



## EggPlant Reference Manual



Copyright 2011 TestPlant Inc.  
Eggplant Reference Manual

### **Trademarks**

Eggplant, the Eggplant logos, TestPlant, and the TestPlant logo are trademarks or registered trademarks of TestPlant Inc.

*Eggplant Reference Manual, Eggplant: Getting Started, Using Eggplant, SenseTalk Reference Manual, and RiTA Guide* are copyrights of TestPlant Inc.

SenseTalk is a trademark or registered trademark of Thoughtful Software, Inc.

Apple, Mac, Macintosh, Mac OS X, and QuickTime are trademarks or registered trademarks of Apple Computer, Inc.

Windows, and Window XP are trademarks or registered trademarks of Microsoft Corporation.

# Contents

## EggPlant Reference Manual

<b>Preface.....</b>	<b>1</b>
About This Manual .....	1
Help and Additional Information .....	1
<b>The EggPlant Menus .....</b>	<b>2</b>
The EggPlant Menu.....	2
About EggPlant .....	2
Preferences... ..	2
License Agreement .....	2
Service Schedule .....	2
Licenses... ..	2
Services .....	3
Hide EggPlant .....	3
Show All .....	3
Quit EggPlant... ..	3
The File Menu .....	3
New Suite... ..	3
Open Suite.....	3
New Script... ..	4
Open Script.....	4
Open Recent Submenu.....	4
Close .....	4
Save .....	4
Save As... ..	4
Save All .....	4
Revert.....	4
Print... ..	4
The Edit Menu .....	4
Undo.....	5
Redo.....	5
Cut.....	5
Copy .....	5
Paste .....	5
Paste As Plain Text.....	5
Select All .....	5
Complete.....	5
Show Resource .....	5
Find .....	6
Spelling .....	6
Special Characters... ..	6
Make Plain Text.....	7
The Run Menu.....	7
Run Script / Abort Script.....	7
Run Selection.....	7
Animation .....	7
Tracing .....	7
Doctor.....	8
Image Highlighting .....	8
Debug Script .....	8

Debug Selection .....	8
Pause Script / Continue Script .....	9
Step Into .....	9
Step Over .....	9
Step Out .....	9
Add Breakpoint / Remove Breakpoint .....	9
Remove All Breakpoints .....	9
The Connection Menu .....	9
Connection List .....	9
Connect .....	10
Disconnect .....	10
Check Availability .....	10
Add Connection .....	10
Edit Connection .....	10
Remove .....	10
The Control Menu .....	10
Enter Capture Mode / Enter Live Mode .....	10
Capture Image .....	10
Use Image .....	11
Find Text .....	11
Script Command submenu .....	11
Control Panel .....	11
Capture Screen .....	11
Refresh Display .....	11
Record Movie / Stop Movie .....	11
Send... Escape Codes and Function Keys .....	11
The Window Menu .....	11
Close Window .....	11
Zoom Window .....	12
Minimize Window .....	12
Bring All to Front / Arrange In Front .....	12
Hide Toolbar / Show Toolbar .....	12
Customize Toolbar .....	12
Active Connection .....	12
Run Window .....	12
List of Open Windows .....	13
The Help Menu .....	13
EggPlant Help .....	13
Tutorial Answer Suite .....	13
EggPlant Examples .....	13
Release Notes .....	13
Submit Question... , Report Bug... , Request Feature... .....	13
<b>The Connection List .....</b>	<b>14</b>
Status .....	14
Name .....	15
Host .....	15
Color .....	15
Port .....	15
SSH Host .....	15
SSH User .....	15
Activity .....	15
Adding or Editing a SUT in the Connection List .....	15
Changing Color Depth .....	16

Opening Secure Connections .....	16
Connecting to a SUT with SSH .....	16
Setting up SSH in the SUT's VNC Server Application .....	17
Opening a VNC Connection with a SUT .....	17
Closing a VNC Connection with a SUT .....	18
<b>The Viewer Window .....</b>	<b>19</b>
Live Mode .....	19
Capture Mode .....	20
The Capture Area .....	20
The Hot Spot .....	20
Capturing Images .....	20
Full-Screen Control .....	20
The Viewer Window Toolbar .....	21
Enter Capture Mode / Enter Live Mode .....	21
Full Size/ Scale to Fit .....	21
Capture Image .....	21
Use Image .....	21
Find Text .....	22
Click, DoubleClick, RightClick, MoveTo, Drag, Drop .....	22
Add Image .....	22
WaitFor .....	22
Wait .....	22
TypeText .....	22
Log .....	23
Comment .....	23
Record Movie/Stop Movie (currently Mac OS X only) .....	23
Capture Screen .....	23
Refresh Screen .....	24
Abort Script .....	24
Cursor Location .....	24
Customize .....	24
<b>The Image Panels .....</b>	<b>25</b>
Capture Panel .....	25
Image Name Field .....	25
Where Pop-Up Menu .....	25
Creating a New Folder .....	25
Making an Image Collection .....	25
Image Well .....	25
Search Type Pop-Up Menu .....	25
Command/Function Pop-Up Menu .....	26
Max Wait Field .....	26
Cancel Button .....	27
Save Button .....	27
The Use Image Panel .....	27
File Browser .....	27
Command/Function Pop-Up Menu .....	27
Max Wait Field .....	27
Cancel Button .....	27
Insert Button .....	27
The Find Text Panel .....	27
Generated Text Platforms .....	28

<b>The Script Editor.....</b>	<b>31</b>
Creating and Editing Scripts.....	31
Auto-completion .....	31
Auto-indenting .....	32
Colorization .....	32
menu features .....	32
Dynamic Breakpoints .....	32
The Script Editor Toolbar .....	32
Save .....	32
Run Script / Abort Script.....	33
Run Selection.....	33
Line Jump Field.....	33
Add Comment/ Comment/Uncomment .....	33
Insert Pull-Down Menu .....	33
Handler Pop-Up Menu .....	34
Show Suite .....	34
Show Resource .....	34
Show Results .....	34
Customize .....	34
Add Image.....	34
Find Text .....	34
Colors, Fonts, and Print .....	34
<b>The Run Window .....</b>	<b>35</b>
Script Display area .....	35
Dynamic breakpoints .....	35
Script frame Pop-up Menu .....	36
The Log Area.....	36
The Ad-hoc do box .....	36
The Run Window Toolbar .....	37
Run Script/Abort Script.....	37
Run Selection.....	37
Pause / Continue .....	37
Step Into .....	38
Step Over .....	38
Step Out .....	38
Animation Pop-Up Menu .....	38
Tracing Pop-Up Menu .....	39
Edit Script.....	39
Show Suite .....	39
Show Results .....	40
Customize .....	40
<b>The Suite Editor.....</b>	<b>41</b>
The Scripts pane .....	41
Searching for a Script.....	41
Creating a new script .....	42
Deleting scripts.....	42
Opening scripts .....	42
Running a script .....	42
Info .....	42
The Images pane .....	42
Adding Images and Image Folders .....	43
Searching for Images .....	43

Deleting Images and Image Folders .....	43
Creating New Image Folders .....	44
The Image Doctor .....	44
Thumbnail view .....	45
Info .....	45
Deleting Images or Image Folders .....	46
The Results pane .....	46
The Script Name/Run Date List .....	47
Deleting Script Names and Run Dates .....	47
The Log Area .....	48
Log Area resource links .....	49
Additional log information in the text column .....	49
Log Area colorization .....	49
Searching for Log Entries .....	49
Action Drop-Down Menu .....	50
The Schedule pane .....	50
Adding Scripts to the Schedule .....	51
Providing a Script's Connection Details .....	51
Changing the Run Order .....	51
Duplicating Scripts in a Schedule .....	52
Setting Dependencies .....	52
Disabling and Removing Scripts from the Schedule .....	52
Running the Schedule Manually .....	52
Running the Schedule at a Particular Time .....	53
The Helpers pane .....	53
Adding a Helper Suite .....	54
Using a Relative Path for a Helper Suite .....	54
Removing a Helper Suite .....	54
The Settings pane .....	54
Suite Version .....	55
Suite Description .....	55
Results Directory .....	55
<b>Licenses and Preferences .....</b>	<b>56</b>
The License Registry panel .....	56
Entering a New License .....	56
Removing a License .....	56
Requesting a Free Trial .....	56
General preferences .....	56
On Startup .....	56
Connection List .....	56
Other Options .....	57
Default Suite Directory .....	57
Viewer Window preferences .....	57
Live Mode Toggle Key Pop-Up Menu .....	57
Live Mode Cursor Pop-Up Menu .....	57
Mouse Scroll Wheel Pop-Up Menu .....	58
Mouse Right Click Key and	
Mouse Middle Click Key .....	58
Other Live Mode preferences .....	58
Capture Mode .....	58
VNC preferences .....	58
Reverse Connections .....	58
VNC Encodings .....	59

Additional VNC Features.....	59
Script preferences .....	59
When a script is run.....	59
When a script is modified.....	59
Reload breakpoints from last save.....	60
Allow Text Drag and Drop.....	60
Default Script Font .....	60
Script Log Font.....	60
The Indentation pane .....	60
The Colorization pane .....	61
Enable syntax coloring .....	61
Run Option preferences .....	61
The Mouse pane .....	61
The Keyboard pane.....	62
The Screen pane.....	62
Image Search Time .....	62
The System pane .....	63
Text preferences .....	63
Text Engines.....	64
Sharing Text Platforms with Other Users .....	65
Mail preferences.....	65
SMTP Server.....	66
Authentication .....	66
User Name and Password .....	66
<b>EggPlant Commands and Functions.....</b>	<b>67</b>
Command and Function basics.....	67
Syntax Guidelines .....	67
Data types used in EggPlant Commands and Functions .....	67
Numbers.....	67
Coordinates .....	68
Strings .....	68
Property Lists .....	68
Image references .....	68
Image Name.....	68
Image Collection .....	69
Image Property List.....	69
Text Property List .....	70
Mouse Commands and Functions .....	71
Click Command.....	71
DoubleClick Command .....	71
RightClick Command .....	72
MouseButtonDown, MouseButtonUp Commands.....	72
ScrollWheelDown, ScrollWheelUp Commands.....	72
MoveTo Command .....	72
MoveToEach Command.....	72
Drag Command.....	73
Drop Command.....	73
DragAndDrop Command.....	73
MouseLocation() Function.....	73
Text Commands and Functions.....	73
CaptureTextImage Command .....	74
KeyDown Command .....	74
KeyUp Command.....	74



ReadTable() Function .....	74
ReadText() Function .....	74
RemoteClipboard() Function .....	75
SetRemoteClipboard Command .....	76
TypeText Command .....	76
Examples: Generating modifiers and special keys on the SUT .....	76
Image-searching Commands and Functions .....	77
Wait Command .....	77
WaitFor Command .....	77
WaitForAll Command .....	77
RefreshScreen Command .....	78
ImageFound() Function .....	78
ImageLocation() Function .....	78
EveryImageLocation() Function .....	78
Image Information Functions .....	78
ImageInfo() Function .....	79
ImageHotSpot() Function .....	79
ImageSize() Function .....	79
ImageColorAtLocation() Function .....	80
Found-Image Information Functions .....	80
FoundImageInfo() Function .....	80
FoundImageLocation() Function .....	81
FoundImageName() Function (Deprecated) .....	81
FoundImageNumber() Function .....	81
Script Commands and Functions .....	81
PauseScript Command .....	81
Run Command .....	82
RunWithNewResults Command .....	82
OpenSuite and CloseSuite Commands .....	82
OpenSuites() Function .....	82
SetOption and SetOptions Commands .....	83
GetOption(), GetOptions() Functions .....	83
Hide RunWindow, Show RunWindow Commands .....	83
Hide RemoteWindow, Show RemoteWindow Commands .....	83
RunningFromCommandLine () Function .....	84
TraceScreen On/Off Command .....	84
Reporting Commands .....	84
Log Command .....	84
LogWarning Command .....	84
LogError Command .....	84
ScriptResults() Function .....	85
SendMail Command .....	85
EggPlantVersion Function .....	86
CaptureScreen Command .....	86
ColorAtLocation() Function .....	86
StartMovie Command (currently Mac OS X only) .....	87
StopMovie Command (currently Mac OS X only) .....	87
SUT Commands and Functions .....	87
AllConnectionInfo() Function .....	87
Connect Command .....	87
ConnectionInfo() Function .....	88
Disconnect Command .....	88
HighlightRectangle command .....	89
RemoteScreenRectangle() function .....	89
RemoteScreenSize() function .....	89

<b>Global Properties.....</b>	<b>90</b>
Using Global Properties with SenseTalk Commands .....	90
Using Global Properties with SetOption and SetOptions .....	90
GetOption() and GetOptions() Functions .....	90
EggPlant Global Properties .....	91
The CollectionFilter .....	91
The ImageDoctor .....	91
The InitialSuites.....	91
The FinalSuites .....	91
The SearchRectangle .....	92
The TextPlatforms .....	92
The CurrentTextPlatform .....	93
The DefaultUseMarkup .....	93
The DefaultTextStyle .....	93
The RemoteClipboard .....	93
The RepositionPoint.....	94
The ScriptLogging .....	94
The ScriptAnimation .....	94
The ScriptTracing .....	95
The CommandLineOutput.....	95
Run Option Global Properties .....	95
The RemoteWorkInterval .....	95
The ImageSearchTime .....	95
The ImageSearchCount .....	96
The ImageSearchDelay .....	96
The KeyDownDelay .....	96
The NextKeyDelay .....	96
The ReadTextSettings.....	96
The SendShiftForCaps.....	97
The MouseClickDelay .....	97
The MouseDoubleClickDelay .....	97
The MouseMoveSpeed .....	97
The MouseDragSpeed .....	98
The MouseMoveDelay .....	98
The MouseMoveMode .....	98
The ShouldRepositionMouse .....	98
The StandardImageTolerance/ PrecisImageTolerance .....	99
The ForceScreenRefresh .....	99
<b>Running from the Command Line.....</b>	<b>100</b>
<b>Running Scripts from the Command Line .....</b>	<b>101</b>
Passing Parameters .....	101
Runscript Options.....	101
Launching the EggPlant GUI from the Command Line .....	102
Command Line Options .....	102

<b>Appendix A: EggPlant Commands .....</b>	<b>104</b>
Eggplant Commands .....	104
<b>Appendix B: EggPlant Functions.....</b>	<b>106</b>
EggPlant Functions .....	106
<b>Appendix C: TypeText Keywords .....</b>	<b>107</b>
TypeText Keywords .....	107
<b>Appendix D: Global Properties .....</b>	<b>109</b>
EggPlant Global Properties .....	109
Run-Option Global Properties .....	110
<b>Appendix E: Property Lists .....</b>	<b>111</b>
CaptureScreen Properties .....	111
Connection Properties .....	111
Image Properties .....	112
Results Properties .....	112
SendMail Properties .....	113
TextPlatform Properties .....	113
Text Properties .....	114
TextStyle Properties .....	115
<b>Appendix F: Pango Markup Tags.....</b>	<b>116</b>
Pango Markup Tags (Currently Mac OS X only) .....	116

# Preface

## About This Manual

---

*The EggPlant: Reference Manual* is designed to provide quick information about the EggPlant interface and script elements.

**Appendices A-E** contain comprehensive lists of the EggPlant script elements, suitable for printing.

## Help and Additional Information

---

The following manuals are available through the EggPlant Help viewer and the downloads page of the [TestPlant web site](#).

*The EggPlant: Reference Manual* describes the EggPlant interface and scripting processes, and the SenseTalk commands, functions, and global properties that are unique to EggPlant.

*Using EggPlant* is a collection of articles that cover a wide range of EggPlant topics.

*EggPlant Tutorials* is a series of five tutorials that introduce the scripting environment and often-used commands and functions.

*The SenseTalk Reference Manual* is a comprehensive guide to the SenseTalk scripting language used in EggPlant.

For EggPlant updates, news, discussion forums, and all available support resources, please visit [TestPlant support](#).

# The EggPlant Menus

This section describes the menus that are always present in the EggPlant menu bar:

- [The EggPlant menu](#)
- [The File menu](#)
- [The Edit menu](#)
- [The Run menu](#)
- [The Connection menu](#)
- [The Control menu](#)
- [The Window menu](#)
- [The Help menu](#)

## The EggPlant Menu

---

The EggPlant menu contains version and license information, preferences, and display options.

### About EggPlant

*About EggPlant* displays your current EggPlant version number, followed by credits for the dedicated and talented EggPlant development team.

### Preferences...

*Preferences* opens the *Preferences* panel, which contains settings for advanced features in EggPlant. Choose *Preferences* to fine tune your test environment. (For more information, see [Licenses and Preferences.](#))

### License Agreement

*License Agreement* brings up a document describing the conditions under which you are licensed to use EggPlant.

### Service Schedule

*Service Schedule* describes the EggPlant Maintenance Service agreement.

### Licenses...

*Licenses* opens the License Registry, a table that provides information on the specific EggPlant licenses available and those in use on the network. (For more information, see [Licenses and Preferences.](#))

## Services

The *Services* submenu provides access to services offered by other applications on your computer. These may or may not be useful to EggPlant.

## Hide EggPlant

*Hide EggPlant* hides the EggPlant interface on your display, and shows the program you used most recently. To return to EggPlant, click the *EggPlant* icon in the Dock.

## Hide Others

*Hide Others* hides the windows of all of the other programs you are currently running. This is helpful for reducing on-screen clutter when you are working with EggPlant. To see a hidden program again, click its icon in the Dock, or choose *EggPlant > Show All*.

## Show All

*Show All* reveals the programs you have hidden with Hide Others. (See [Hide Others](#), above.)

## Quit EggPlant...

If a script is running, *Quit EggPlant* displays a confirmation dialog before quitting the program. You can choose to *Quit Now*, *Quit After Next Script Run*, or *Cancel*. If there is no script running, you can bring up this dialog by holding down Option while you quit.

If you click *Quit After Next Script Run*, EggPlant delays the quit until the next script run is complete, then quits without further confirmation. (To cancel a delayed quit, choose *Quit EggPlant* again, and click *Cancel* in the confirmation dialog.)

## The File Menu

---

The *File* menu allows you to manage test suites and scripts; there are menu items that create, open, close, save, revert, and print.

## New Suite...

*New Suite* creates a new test suite in the Suite Editor. (For more information, see [The Suite Editor](#).)

**Note:** You can name a default directory for new suites in the General preferences panel.

## Open Suite...

*Open Suite* brings up an Open dialog in which you can select an existing suite to open.

## New Script...

*New Script* creates opens the Script Editor for a new script for the active suite. The new script appears in the Scripts pane of the Suite Editor.

## Open Script...

*Open Script* brings up an *Open* dialog in which can select an existing script to open.

## Open Recent Submenu

*Open Recent* displays a submenu of suites and scripts that have recently been opened. Choose a suite or script from the submenu to open it.

To delete the list of files in the *Open Recent* submenu, choose *File* menu > *Open Recent* > *Clear menu*.

## Close

The *Close* menu item closes the active window.

**Note:** To close all EggPlant windows at once, hold down *Option* as you close a window.

## Save

If the Script Editor is the active window, the *Save* menu items saves the active script. If a Suite Editor is the active window, the *Save* menu item saves all open scripts in the active suite.

## Save As...

The *Save As* menu item saves the active script under a new name that you specify.

## Save All

The *Save All* menu item saves all open scripts and suites.

## Revert

*Revert* reloads the last saved version of the active script in the Script Editor.

## Print...

The *Print* menu item prints the active script.

## The Edit Menu

---

The *Edit* menu contains a standard menu items for editing and finding text in your scripts.

## Undo

*Undo* reverses the last change you made to the text in the Script Editor. You can undo and redo many steps, in the order in which they were performed.

## Redo

*Redo* reverses the *Undo* menu item.

## Cut

*Cut* removes selected text from your script and stores it in the computer's clipboard. After you cut text, you can paste it elsewhere using the *Paste* menu item.

## Copy

The *Copy* menu item copies selected text to the computer's clipboard without removing it from its original location. After you copy text, you can paste it using the *Paste* menu item.

## Paste

*Paste* inserts cut or copied text from the computer's clipboard into a script.

**Note:** When you have a VNC connection open, you can paste text between the local computer and the SUT.

## Paste As Plain Text

*Paste As Plain Text* works just like the *Paste* menu item, except that it removes formatting such as font, style, and color before pasting the text.

## Select All

The *Select All* menu item selects every selectable item in the active window, or all of the text in a text field.

## Complete

When you have typed at least two characters in the Script Editor, *Complete* attempts to finish your current word using variables, commands, and names of scripts or images from your current suite. If there is a single match, the word is completed. If there are multiple matches, you can select the appropriate word from a list of options.

To cancel the Complete command without selecting any of the options, press the Escape key.

## Show Resource

*Show Resource* opens the script or image selected in your text. (Option-click is a shortcut for *Show Resource*.)



## Find

*Find* displays a submenu of the following commands that search for text in your scripts:

- **Find.** The Find menu item opens the Find panel.
- **Find Next.** Find Next searches forward in the current document and displays the next occurrence of the current Find term.
- **Find Previous.** Find Previous searches backward in the current or previous document and displays the last occurrence of the Find term.
- **Use Selection for Find.** Use Selection for Find takes the selected text in the Script Editor as the current Find term.
- **Jump to Selection.** Jump to Selection scrolls through the current window to display the selected text. If no text is selected, Jump to Selection displays the current insertion point.

## Spelling

*Spelling* contains a submenu of standard items for checking spelling in the Script Editor:

- **Spelling.** The Spelling menu item checks the spelling in the current window (or selected text) and opens a dialog that suggests corrections. You can step through the document using the Find Next button to locate other possible misspellings.
- **Check Spelling.** The Check Spelling menu item checks the spelling in the current window and highlights the next suspected misspelling. (Command-click and right-click are shortcuts for checking the spelling of selected text.)
- **Check Spelling As You Type.** If Check Spelling As You Type is turned on, the Script Editor checks your spelling as you type, underlining suspected errors. (Misspelled words may only be marked briefly if you have the “Update colors continuously while typing” option turned on in Script Editor preferences.)

**Tip:** The spelling features are more useful if you add EggPlant scripting commands to your system dictionary. Otherwise, the commands are treated as misspelled words.

## Special Characters...

The *Special Characters* menu item opens a panel that allows you to insert special Unicode characters into your script.

**Note:** Typing an international character in your script does not guarantee that it can be reproduced on the SUT. The ability to send particular characters to a SUT depends on the capabilities of the SUT’s operating system and VNC server.

**Example #1:** Your SUT and its VNC server both support unicode characters, so any unicode character in your script can be typed on the SUT with no additional steps.

**Example #2:** Your SUT uses an American keyboard layout, and when you type Asian characters, you use an input method editor (IME). EggPlant would *also* have to use the IME to send Asian characters to this SUT.

**Example #3:** Your SUT supports unicode characters, but its VNC server does not. It may be possible to find a VNC server that supports unicode characters; otherwise, EggPlant would need to use an IME to produce international characters on the SUT.

## Make Plain Text

This option converts older rich text scripts to plain text.

## The Run Menu

---

The *Run* menu contains controls for running and debugging scripts. (For more information, see [The Run Window](#).)

### Run Script / Abort Script

The Run Script menu item runs the script in the active Script Editor or Run window. While a script is running, the *Run Script* menu item becomes *Abort Script*.

**Tip:** To start the script in a paused state for debugging, hold down Option as you choose Run Script.

### Run Selection

The *Run Selection* menu item runs only the selected part of a script in the active Script Editor or Run window. (This menu item is only available when there is selected text in the active window.)

**Tip:** To start the selected part of a script in a paused state for debugging, hold down Option as you choose Run Selection.

## Animation

*Animation* highlights the current line of code in the Script Display area of the Run window as the script is running.

The script animation setting can be changed while a script is running, with immediate effect; it can also be modified temporarily by scripts as they run. (For more information, see [the ScriptAnimation in Global Properties](#).)

- **Animation Off:** When you choose Animation Off, the Display area of the Run window does not update as the script runs; only the initial handler for the script is shown, unless the execution is paused.
- **Animate Calls:** When you choose Animate Calls, the Run window shows each script or handler as it is called, but it does not highlight each line of the script.
- **Animate All:** When you choose Animate all, the Run window highlights each line of each script or handler as it is run.

**Note:** Animations cause scripts to execute more slowly, because EggPlant is doing more work at the display level. To improve script execution speed, turn off all animations when you are not debugging.

## Tracing

Script tracing causes EggPlant to “echo” information about what the script is about to execute to the Log Area in the Run window. This is similar to script animation, but it allows you to review *all* of the code that is executed during the script run, interspersed with logged output. This can provide more context when debugging a script.

The tracing setting can be changed while a script is running, with immediate effect; it can also be modified temporarily by scripts as they run. (See [the ScriptTracing in Global Properties](#).)

- **Tracing Off:** When Tracing Off is selected, only standard log messages are displayed in the Log Area of the Run window.
- **Trace Calls:** Trace Calls causes the Log Area of the Run window to display a message as it enters and leaves each script or handler.
- **Trace All:** Trace All causes the Log Area of the Run window to display every line of each script or handler prior to executing it.

**Note:** Tracing cause scripts to execute more slowly, because EggPlant is doing more work at the display level. To improve script execution speed, turn off tracing when you are not debugging.

## Doctor

The *Doctor* submenu contains options for the way the Image Doctor works during script execution:

- **Auto:** The Image Doctor automatically attempts to correct image failures and continue the script execution. (The initial image failure is logged as a warning.)
- **Manual:** The Image Doctor panel opens when an image fails; you can choose to proceed with a corrected image, or allow the script to fail.
- **Off:** The Image Doctor is not used during script execution.

## Image Highlighting

This menu item toggles image highlighting on (checked) or off (unchecked). When image highlighting is turned on, images are highlighted in the Viewer window as they are found during a script execution.

## Debug Script

*Debug Script* runs the script in the active Script Editor or Run window, starting in a paused state. This allows you to step through the script and follow its execution closely (The next line to be executed is highlighted in the Script Display area of the Run window.)

While you are debugging, you can use the toolbar buttons in the Run window to step through the script, or allow it to continue at a normal rate. (For more information, see [The Run Window](#).)

**Note:** You can also use Debug Script by Option-clicking the Run Script button in the Script Editor toolbar.

## Debug Selection

*Debug Selection* runs only the selected part of a script in the active Script Editor or Run window, starting in a paused state. This allows you to step through the script and follow its execution closely (The next line to be executed is highlighted in the Script Display area of the Run window.)

While you are debugging, you can use the toolbar items on the Run window to step through the script, or allow it to continue at a normal rate. (For more information, see [The Run Window](#).)

**Note:** You can also use Debug Selection by Option-clicking the Run Script button in the Script Editor toolbar.

## Pause Script / Continue Script

The *Pause/Continue* menu items pause and continue the current script execution. This allows you to examine the state of the script and interact with the Viewer window in Live Mode.

## Step Into

*Step Into* executes the next line of a script, then pauses. If this line calls another script or handler, the script “steps into” the first line of the called script or handler before pausing again.

*Step Over* performs a similar function, with one exception: If the line calls another script or handler, *Step Over* executes the called script or handler entirely before pausing.

## Step Over

Like *Step Into*, *Step Over* executes the next line of the script, then pauses. However, if this line calls another script or handler, that script or handler runs in its entirety, “stepping over” it to the next line of original script before pausing again.

During the execution of the script or handler that is stepped over, you can click *Pause* to regain control of the execution before the *Step Over* action has finished.

## Step Out

*Step Out* executes all of the remaining lines in the current script or handler, then pauses. (Using *Step Out* on an initial handler is the same as clicking *Continue*.)

During the execution of the code being stepped out of, you can click *Pause* to regain control before the *Step Out* action has finished.

## Add Breakpoint / Remove Breakpoint

The *Set Breakpoint/Remove Breakpoint* commands insert or remove a dynamic breakpoint in the selected line of a script. (For more information, see [Dynamic Breakpoints in The Script Editor](#).)

## Remove All Breakpoints

The *Remove All Breakpoints* command removes all breakpoints in all of your currently open scripts.

## The Connection Menu

---

The *Connection* menu contains items that allow you to open and close VNC connections to SUTs. (For more information, see [The Connection List](#).)

## Connection List

The *Connection List* menu item opens the Connection List, in which you can manage and open VNC connections to your SUTs. (For more information, see [The Connection List](#).)

## Connect

The *Connect* menu item attempts to establish a VNC connection with the SUT selected in the Connection List.

## Disconnect

*Disconnect* closes the VNC connection with the SUT selected in the Connection List.

## Check Availability

*Check Availability* updates the availability status of SUTs in the Connection List.

## Add Connection...

*Add Connection* opens the Add/Edit panel in the Connection List, allowing you to add a new SUT to the Connection List.

## Edit Connection...

*Edit Connection* opens the Add/Edit panel in the Connection List, allowing you to edit the selected SUT.

## Remove

*Remove* deletes the selected SUT from the Connection List.

## The Control Menu

---

The *Control* menu contains many of the features that are in the Viewer window toolbar. (For more information on these menu items, see [The Viewer Window](#).)

## Enter Capture Mode / Enter Live Mode

Enter Capture Mode disables your remote control of a SUT, so you can use your local EggPlant tools to capture images in the Viewer window. Capture Mode dims the Viewer window, making the selected Capture Area stand out at normal brightness. (For more information on capturing images, see [Capture Mode, in The Viewer Window](#).)

*Enter Live Mode* restores your remote control of the SUT.

To toggle between Capture Mode and Live Mode quickly, press Command. (You can change the toggle key in Viewer preferences.)

## Capture Image...

When the Viewer window is in Capture Mode, *Capture Image* takes a snapshot of the Capture Area.

## Use Image...

*Use Image* brings up an Open panel in which you can select a saved image to use in your script.

## Find Text...

*Find Text* (formerly *Text Image*) allows you to search for text based on content and formatting properties.

## Script Command submenu

The *Script Command* submenu contains common EggPlant script commands and functions. Each menu item inserts the indicated command into the current script. (For more information, see [The Viewer window Toolbar in The Viewer Window](#).)

## Control Panel

The Control panel allows you to enter Full-Screen Mode, viewing the SUT in place of your own desktop.

## Capture Screen

*Capture Screen* takes a snapshot of the entire Viewer window of your SUT. (For more information, see [Capture Screen, in The Viewer Window](#).)

## Refresh Display

*Refresh Screen* requests a full-screen update from the SUT. This is useful for cleaning up occasional artifacts (drawing errors) during scripting.

## Record Movie / Stop Movie

*Record movie* opens a *Save* dialog in which you select a location to save your QuickTime movie. Recording starts when you click the *Save* button. (For more information, see [Record movie, in The Viewer window](#).)

## Send... Escape Codes and Function Keys

These items allow you to send commands to the SUT that would otherwise be intercepted by the EggPlant computer. For example, if you tried to type Control-Alt-Delete to force quit an application on your SUT, your EggPlant computer would intercept that command, and quit EggPlant. Instead, you could choose the *Send Control Alt Delete* menu item, and the command would bypass your local system.

## The Window Menu

---

The *Window* menu contains items that allow you to manage EggPlant windows and toolbars.

## Close Window

The *Close* menu item closes the active window.

**Note:** To close all EggPlant windows at once, hold down Option as you close a window.

## Zoom Window

For most windows, *Zoom Window* enlarges the active window to the size of your display, or returns it to its previous size. (The exception is the Viewer window, which never grows larger than its contents.)

## Minimize Window

*Minimize Window* sends the active window to the Dock. (You can minimize all EggPlant windows at once by holding down Option as you minimize a window.)

To see the window again, click it in the Dock, or choose it in the *Window* menu.

## Bring All to Front / Arrange In Front

*Bring All To Front* moves all open EggPlant windows in front of windows from other programs.

You can arrange the EggPlant windows in a cascading sequence by holding down the *Option* key as you choose Bring to Front.

## Hide Toolbar / Show Toolbar

*Hide Toolbar* causes the toolbar of the active window to disappear. If the active window's toolbar is already hidden, the action becomes *Show Toolbar*. (The oblong toolbar button at the right end of the title bar is a shortcut to Hide/Show Toolbar.)

## Customize Toolbar

*Customize Toolbar* brings up a panel in which you can select the buttons that are available in the toolbar of the active window.

To add a button to the toolbar, drag the button from the Customize panel to the toolbar;

To remove a button from the toolbar, drag it off of the toolbar.

While the Customize panel is open, you can also drag buttons to different locations within the toolbar.

## Active Connection

The *Active Connection* menu item brings up the Viewer window of the SUT you are currently working with. If your VNC connection has been closed, you can still bring up the Viewer window showing the last known state of the SUT.

## Run Window

The *Run Window* menu item brings the Run window in front of all other windows. If the Run window is minimized or closed, choosing *Run Window* opens it.

## List of Open Windows

The bottom of the *Window* menu contains a list of all open EggPlant windows except the Active Connection and Run windows (which have their own entries in the *Window* menu, above.)

Choosing any window brings it to the front of all other windows. This is useful for finding windows on a cluttered desktop.

## The Help Menu

---

The *Help* menu contains the EggPlant Help Viewer, as well as channels through which you can provide feedback about EggPlant.

## EggPlant Help

The EggPlant Help Viewer allows you to search across all of the EggPlant help documents: Using EggPlant, EggPlant Reference Manual, Getting Started, EggPlant Tutorials, and the SenseTalk Reference Manual.

**Tip:** To show the EggPlant manuals in the Mac OS X Finder, hold Option as you choose Help menu > EggPlant Documentation.

## Tutorial Answer Suite

The tutorial answer suite contains the scripts and images used in EggPlant Tutorials.

## EggPlant Examples

[EggPlant Examples](#) is a discussion forum where you can also post questions and answers for other EggPlant users and TestPlant support.

## Release Notes

The *Release Notes* document contains information about the current release of EggPlant, including changes that have been made since the previous release.

When you update to a new version, please check the Release Notes to read about significant changes to the software.

## Submit Question... , Report Bug... , Request Feature...

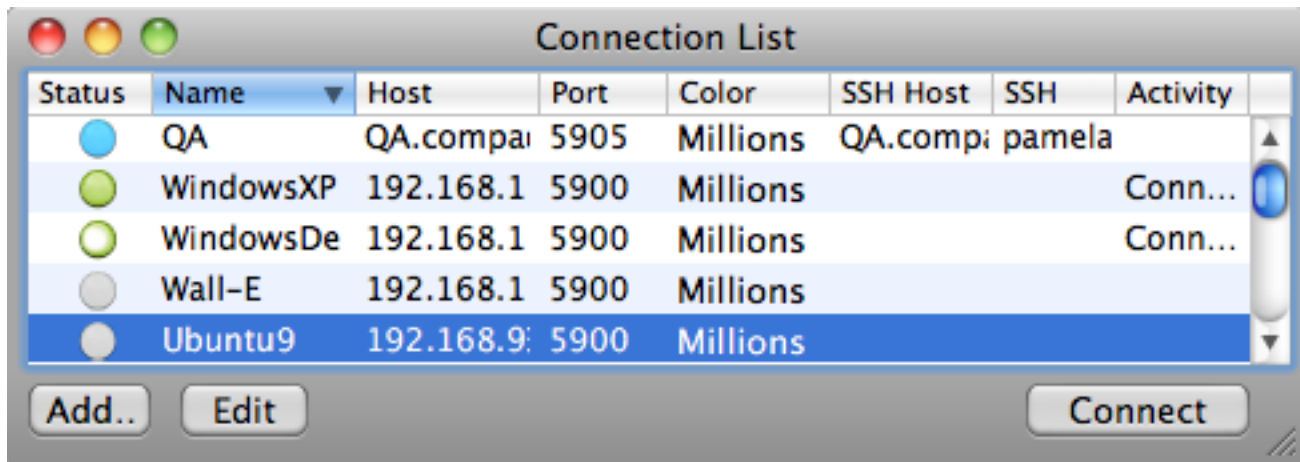
These three menu items open an EggPlant Support Request panel.



## The Connection List

The Connection List contains the names and other information about all of the SUTs you can control through EggPlant. (For more information, see *EggPlant: Getting Started*.)

### *The Connection List*



### *The Connection List*

## Connection List Data

The columns in the Connection List contain the following information for each SUT:

### Status

The *Status* column indicates the SUT's availability, denoted by a color-coded dot. To receive a status update, choose Connection menu > Availability.

### Connection List status colors

Color		Meaning
	Green	Connected
	Blue	Available
	Gray	Unavailable
	Yellow	Unknown
	Partial Yellow	Connecting
	Red	Error
	Gold Halo	Active Connection

## Name

The *Name* column refers to the display name a SUT sends to EggPlant. Before EggPlant connects to a SUT, the Name column displays the same information as the *Host* column.

**Tip:** To insert the name of a SUT into a script quickly, drag it from the Connection List to the Script Editor. To insert a connection property list into the script, Option-drag the SUT to the Script Editor.

## Host

The *Host* column displays the SUT's host name or IP address.

## Color

The *Color* column displays the setting you choose in the *Color Depth* menu.

## Port

The *Port* column displays the number of the port on which the SUT listens for VNC connections.

## SSH Host

The *SSH Host* column displays the host name or IP address of the computer that hosts an SSH tunnel for the SUT's secure connections.

## SSH User

The *SSH User* column displays the account name with which the SUT user logs into an SSH host.

## Activity

The *Activity* column displays a running description of your VNC connection.

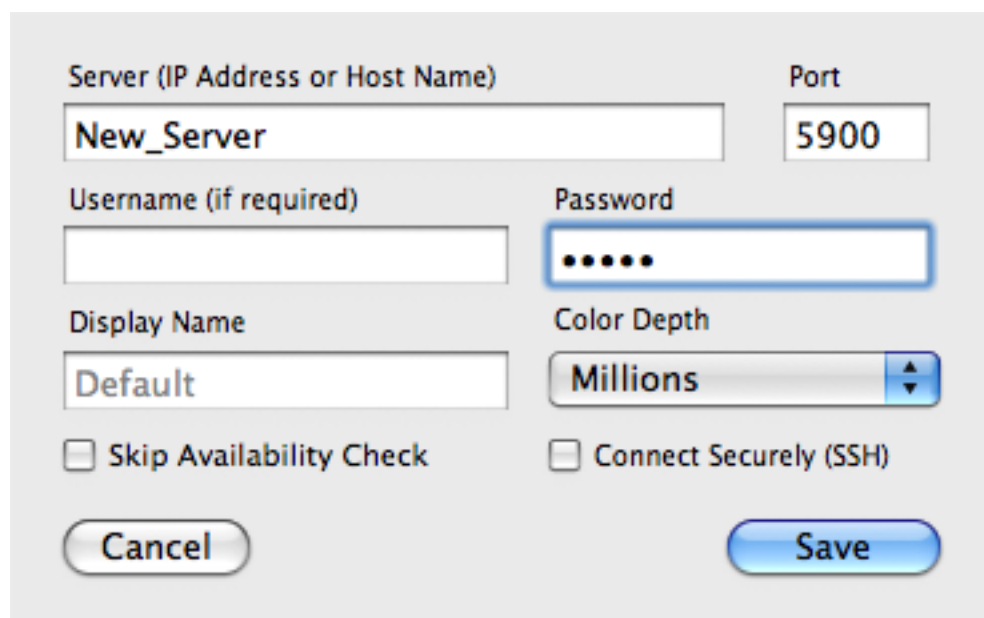
## Adding or Editing a SUT in the Connection List

- 1 In the *Connection* menu, choose Connection List.
- 2 In the Connection List, Click *Add*, or select a SUT and click *Edit*.

Fill in the text fields with the SUT's VNC server information:

- **Server.** Host Name or IP. The remote computer's host name (such as *vine.testplant.com*) or IP address.
- **Port.** The port number of the VNC server (5900-5909).
- **Username (if required).** Your user account on the SUT. (Some VNC servers require your system-level account name and password as authentication.)

- **Password.** The password required by the SUT's VNC server.



*Adding a SUT to the Connection List*

## Changing Color Depth

To increase the speed of your VNC connection, you can decrease the color depth with which the Viewer window draws a SUT.

To increase or decrease color depth, choose a value in the *Color Depth* pop-up menu. (Choose Default to draw the SUT with the same color depth you see on its native display.)

**Note:** If you change the color depth of a SUT, images you captured in the former color depth might not match the SUT in the current Viewer window.

## Opening Secure Connections

Secure Shell (SSH) is a network protocol that uses data encryption to transfer information securely. On Mac OS X, you do not need to install any additional tools to open SSH-encrypted connections in EggPlant,

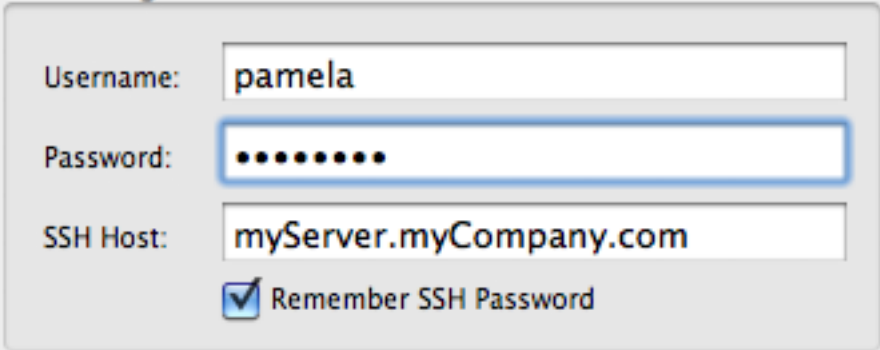
For EggPlant to connect to a SUT securely, the SUT must be able to host an SSH tunnel, or have a secure connection with another computer that can host an SSH tunnel. (You must have a user account on the SSH host computer.)

## Connecting to a SUT with SSH

For EggPlant to connect to a SUT securely, the SUT must be able to host an SSH tunnel, or have a secure connection with another computer that can host an SSH tunnel. (You must have a user account on the SSH host computer.)

## Setting up an SSH Login

- 1 In the Connection List, select the SUT.
- 2 Click Edit.
- 3 Select the "Connect Securely (SSH)" checkbox.
- 4 Fill in the Username and Password fields. (Username and Password refer to your user account on the SSH Host computer.)
- 5 In the "SSH Host" field, enter the IP address or host name of the SSH host computer.
- 6 Select the "Remember SSH Password" checkbox to save the password of the user account on the SSH host; otherwise, you must enter the password whenever you open a connection with the SSH host.



The image shows a dialog box titled "SSH Login". It contains three text input fields: "Username:" with the value "pamela", "Password:" with masked characters "••••••••", and "SSH Host:" with the value "myServer.myCompany.com". Below the "SSH Host" field is a checked checkbox labeled "Remember SSH Password". At the bottom of the dialog are two buttons: "Cancel" on the left and "Save" on the right.

SSH Login panel

## Setting up SSH in the SUT's VNC Server Application

When you use SSH to connect to a SUT, its firewall must allow connections on port 22, the standard port for SSH connections.

For more information, see the SUT's VNC server documentation. (If you are using Vine Server, see the Vine Server User Manual, Allowing VNC Connections.)

## Opening a VNC Connection with a SUT

There are several ways to open a VNC Connection with a SUT:

- In the Connection List, double-click the name of the SUT.
- Select the SUT and click Connect.
- Select the SUT and choose Connection menu > Connect.

**Note:** You can open a VNC connection from within a script by inserting a [Connect command](#).

**Tip:** If you are running a script that switches between SUTs, you can save execution time by keeping both (or all) connections open, rather than disconnecting and re-connecting with each switch.

## Closing a VNC Connection with a SUT

There are two ways to close a VNC Connection manually:

- In the Connection List, select the SUT and click the Disconnect button.
- While the SUT's Viewer window is the key window, choose Connection menu > Disconnect.

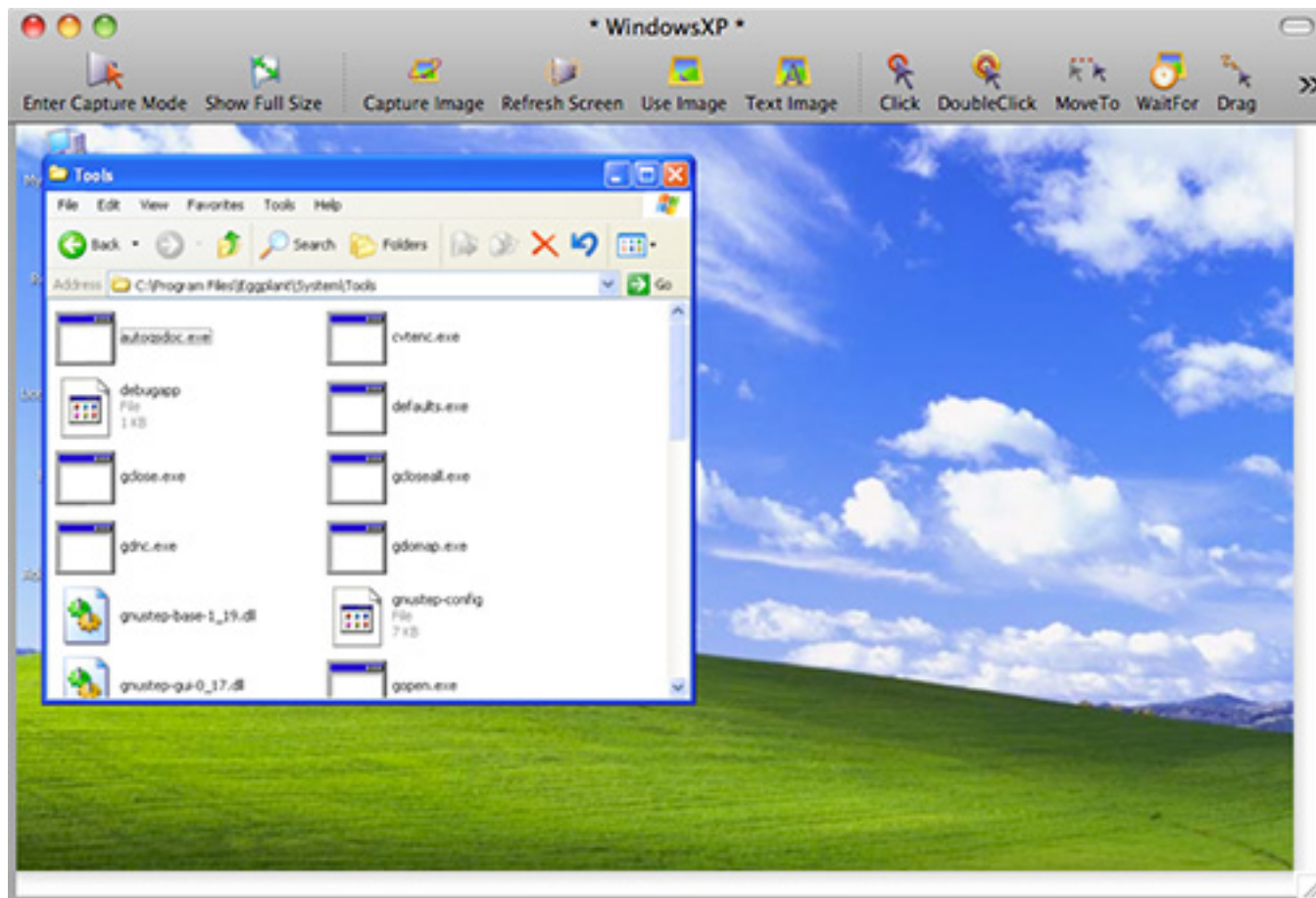
**Note:** Closing a Viewer window does *not* close your VNC connection with the corresponding SUT.

**Note:** You can close a VNC connection from within a script by inserting a [Disconnect command](#).

## The Viewer Window

The Viewer window shows an image of the SUT as you would see it on its own display. In the Viewer window, you can interact with the SUT as a user in Live Mode, and capture images to use in EggPlant scripts in Capture Mode.

**Note:** To toggle between Live Mode and Capture Mode quickly, press Command. (You can select a different key to perform this action in Viewer Window preferences.)



*The Viewer window*

## Live Mode

In Live Mode, you can use your local mouse and keyboard to interact with the SUT as a user. This works just as if you were using the SUT's own keyboard and mouse directly. (If the Viewer window is not your active window, your clicks and keystrokes are applied to your local computer, not the SUT.)

**Note:** In Live Mode, you can also copy and paste text between applications running on the SUT and your local computer.

## Capture Mode

---

In Capture Mode, you can capture images of your SUT and insert commands and functions into your current script. You continue to receive display updates from the SUT, but your mouse and keyboard events are applied to your local computer.

The Viewer window is dimmed in Capture Mode, except for the Capture Area, described below.

### The Capture Area

In Capture Mode, the Capture Area shows the area that is included when you save an image. It is the rectangle shown at normal brightness when the rest of the Viewer window is dimmed.

To move the Capture Area, drag the edges. You can also nudge the Capture Area one pixel at a time by pressing the Arrow keys. (Add the Shift key to move the Capture Area in ten-pixel increments.)

To resize the Capture Area, drag the corners. You can also nudge the size one pixel at a time with Option-Arrow keys. (Add the Shift key to resize in ten-pixel increments.)

**Note:** The Capture Area is drawn with a grey border to make it easier to see on dark backgrounds; you can turn this border off in Viewer window preferences.

### The Hot Spot

The Hot Spot is the point that is clicked when a script executes a `click` command on an image. The Hot Spot is typically marked with red cross hairs; it automatically turns black when it is positioned over a predominantly red image.

To move the Hot Spot in a Capture Area, Command-click or Command-drag the cross hairs. You can also nudge the Hot Spot in one-pixel increments with Command-Arrow. (Adding the Shift key moves the Hot Spot in ten-pixel increments.)

## Capturing Images

To capture an image of the Capture Area, click the *Capture Image* button in the Viewer window toolbar, or choose *Control* menu > *Capture Image*. (For more information, see [The Capture Panel](#).)

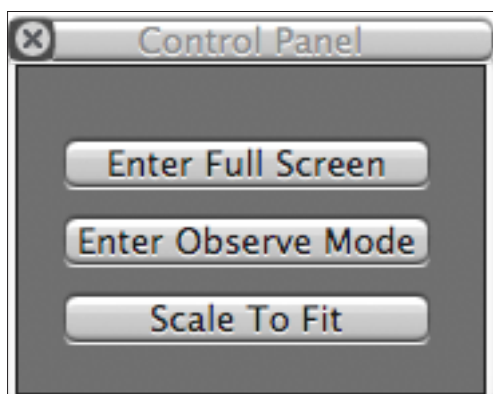
## Full-Screen Control

With full-screen control, you can view the SUT as your entire desktop. A small control panel provides access to the regular Viewer window features.

To open the control panel, choose *Control* menu > *Control* panel.

To customize the buttons displayed on the control panel, right-click in the panel, and choose Viewer window items from the contextual menu.

To return to your own desktop, click the *Exit Full screen* button, or use the keyboard shortcut *Option+Command+1*.



*The full screen control panel*

## The Viewer Window Toolbar

---

Most of the buttons in the Viewer window toolbar are also represented as menu items in the *Control* menu.

The toolbar buttons are described below. Since the toolbar can be customized, you might not see all of the buttons on your own toolbar. (For more information on adding and removing toolbar buttons, see *Customize*, below.)

### Enter Capture Mode / Enter Live Mode

This button toggles the Viewer window between Live Mode and Capture Mode. When you click the *Enter Capture Mode* button, it turns to *Enter Live Mode* button, and vice versa.

**Note:** You can also toggle between Live Mode and Capture Mode quickly by pressing Command. (You can select a different key to perform this action in Viewer window preferences.)

### Full Size/ Scale to Fit

This button toggles between showing the Viewer window at the SUT's full display size, and showing the entire SUT scaled to fit the size of the Viewer window.

**Note:** Switching between the sizing options has no effect on image captures or matching.

### Capture Image

The *Capture Image* button takes a snapshot of the Capture Area. (For more information, see [The Capture Panel](#).)

### Use Image

The *Use Image* button opens the Use Image panel, in which you can select a saved image to use in a script. (Option-clicking a command button also opens a Use Image panel.)

At the bottom of the Use Image panel, there is a pop-up menu of EggPlant commands and functions that you can insert into the script with your selected image. (For more information, see [The Use Image Panel](#).)



## Find Text

The *Find Text* button (formerly the *Text Image* button) opens the Find Text panel, which allows you to search for text based on content and formatting attributes. (Shift-clicking a command button also opens the Find Text panel.) (For more information, see [The Find Text Panel](#).)

## Click, DoubleClick, RightClick, MoveTo, Drag, Drop

These command buttons perform three actions at once:

- 1 They open the Capture panel to save an image of the current Capture Area. (To open the Use Image panel instead of a Capture panel, Option-click the button.)
- 2 They insert their respective command into the active script.
- 3 They perform their respective action on the SUT.

## Add Image

The *Add Image* button takes a snapshot of the Capture Area, and adds it to the line indicated by your insertion point in the Script Editor. (The exact location of the insertion point does not matter, as long as it is in the correct line of the script.) The *Add Image* button is only available if the Viewer window is in Capture Mode.

## WaitFor

The `WaitFor` command causes a script to pause and wait for an image to appear in the Viewer window.

When you click the *WaitFor* button, a Capture panel opens. In the *Maximum Wait* text field, enter the longest period of time (in seconds) that you want the script to wait for the image to appear in the Viewer window.

When you run the script, the script executes the next command as soon as the `WaitFor` image appears. If the image does not appear in the maximum wait time, the test fails.

## Wait

The `Wait` command causes a script to wait for a specified period of time before executing the next command. This way, you can allow an arbitrary length of time for the SUT to complete a task before proceeding with the script.

The *Wait* button opens a dialog in which you must specify the time (in seconds) of the `Wait` command.

**Tip:** The `WaitFor` command has two advantages over the `Wait` command:

- `WaitFor` delays the next action in your script only as long as necessary to locate the appropriate image, while `Wait` delays the next action for a fixed period of time, even if less time is needed.
- The `WaitFor` command returns the time it took for its target image to appear. If you use the `Wait` command, you only know that the image appeared sometime in the Maximum Wait time frame.

## TypeText

The `TypeText` command sends text as keystrokes typed in the Viewer window, allowing you to simulate typing on the SUT.

The *TypeText* button opens a panel in which you can enter your text parameters. When you create a *TypeText* command with this panel, clicking the *Insert* button automatically encloses your text parameters in quotation marks in your script.

**Note:** Text parameters enclosed in quotation marks are typed as literal text on the SUT. Text parameters that are not enclosed in quotation marks are treated as variables.

**Tip:** When the Viewer window is in capture mode, you can start a *TypeText* command by typing the text you want to insert into your script. For more information, see "Recording *TypeText* Commands" in *Using EggPlant*.

## Log

The *Log* command creates a custom entry in a script's execution log.

To insert a *Log* command, click the *Log* button and type an entry into the Log panel. (If you provide multiple parameters, each one is added to the script as a separate entry.)

**Note:** You can see the execution log in the Results pane of the Suite Editor (For more information, see [The Results pane, in The Suite Editor](#).)

## Comment

The *Comment* button opens a panel in which you can type a comment for the active script. The comment is inserted into the script in the form (*\*Comment\**). (For more information, see [Comment in The Script Editor](#).)

## Record Movie/Stop Movie (currently Mac OS X only)

The *Record movie* button opens a *Save* dialog in which you select a location to save your QuickTime movie. Recording starts when you click the *Save* button.

All screen updates from the SUT are recorded in the movie, whether they are caused by a script, your interaction with the Viewer window, or a user operating the SUT directly.

To end the movie, click the *Stop movie* button.

**Note:** The *Record movie* button does not create a script command; to add movie recording to a script, you must edit the script manually.

## Capture Screen

The *Capture screen* button takes a snapshot of the entire Viewer window of your SUT.

Although screen captures in your Images directory appear in the Images pane of the Suite Editor, they are not added to a script when you choose *Capture screen*. Instead, *Capture screen* is useful for other purposes, such as documentation and interface design discussions.

**Note:** The *CaptureScreen* command performs screen captures from within a script, as opposed to the *Capture screen* button, which performs screen captures independently of a script. (For more information, See [CaptureScreen in EggPlant commands and functions](#).)

## Refresh Screen

The *Refresh screen* button requests a full-screen update from the SUT. This is useful for cleaning up occasional artifacts (drawing errors) during scripting.

**Note:** The *Refresh screen* button does not add a command to your script, as opposed to the `RefreshScreen` scripting command. (For more information, See [The RefreshScreen command in EggPlant commands and functions.](#))

## Abort Script

*Abort Script* stops a script in progress. It stops before executing the next line of the script.

## Cursor Location

This field contains screen coordinates for the SUT cursor in the Viewer window. You can move the cursor to a particular location by editing the coordinates in this field.

## Customize

The *Customize* button opens a panel in which you can select the buttons that are available in the Viewer window toolbar.

To add a button to the toolbar, drag the button from the Customize panel to the toolbar;

To remove a button from the toolbar, drag it off of the toolbar.

While the Customize panel is open, you can also drag buttons to different locations within the toolbar.

# The Image Panels

This section describes the panels you use to create images and insert existing images into your scripts.

## Capture Panel

---

The Capture panel is similar to a Save dialog, with some additional script and image controls.

### Image Name Field

The *Image Name* field defaults to a dynamically generated name that is unique to the current suite, beginning with *image0001*. This name may be sufficient for temporary scripts and experimentation, but more descriptive names are easier to follow as you start scripting in earnest.

Image names can contain alphanumeric characters, spaces, and most punctuation marks.

### Where Pop-Up Menu

The *Where* pop-up menu displays the folders in which you can save an image. The default choice is the Images folder of the current suite.

### Creating a New Folder

To create a subfolder within the currently selected folder, click the *New Folder* button.

### Making an Image Collection

To make an Image Collection based upon an existing image, select the image and click the *Make Collection* button. The Image Collection takes on the same name as the image, and the image is automatically moved into the Image Collection.

### Image Well

The central portion of the Capture panel is an image well that displays the captured image.

The Hot Spot in the image is indicated by red cross hairs. To move the Hot Spot, Command-click in the image, or Command-drag the cross hairs.

**Note:** If the captured image is too large for the image well, it is scaled down to fit. The actual image is not affected, but it may appear distorted in the Capture panel.

### Search Type Pop-Up Menu

Search type describes the level of precision EggPlant requires to consider a part of the Viewer window a match for your target image. In the *Search Type* pop-up menu, choose one of the following search types:

## Precise

The Precise search type requires a very high degree of precision to consider an image matched in the Viewer window. This is useful for images with subtle color differences or little contrast.

## Tolerant

The Tolerant search type accepts matches that have slightly different pixel colors caused by background and transparency changes. Tolerant searches are more forgiving than precise searches, so they reduce the chance of a script failing because of a minor variation in rendering.

## Pulsing

The Pulsing search type allows for pixel colors that can change, such as the pulsing buttons in Mac OS X.

## Text

The Text search type allows for dynamic font smoothing (text anti-aliasing) in images that contain text. If EggPlant sometimes fails to locate an image that contains text, try changing the search type to *Text*.

## Text&Pulsing

The Text&Pulsing search type allows for both changing pixel colors and text anti-aliasing.

## The Default Search Type

The default choice in the *Search Type* pop-up menu depends upon the image you are saving. EggPlant attempts to choose the most appropriate search type for each image. (*Tolerant* is usually the best choice; *Text* and *Text&Pulsing* are never default choices.)

**Note:** You can adjust the tolerance of the Precise and Tolerant search types settings in Run Options preferences.

## Command/Function Pop-Up Menu

To insert a command or function into the script with your image, choose the command or function in the *Command/Function* pop-up menu.

To insert the image into the script with no command or function, choose *Capture Image*.

To insert the image into a selected command or function in the script, choose *Add Image*.

## Max Wait Field

To specify the maximum wait time associated with the `wait` or `waitfor` commands, type the time (in seconds) in the *Max Wait* field.

## Cancel Button

To cancel the image capture and close the Capture panel, click the *Cancel* button.

## Save Button

To complete the image capture and close the Capture panel, click the *Save* button.

## The Use Image Panel

---

The Use Image panel is similar to a standard *Open* dialog, with added script controls at the bottom.

You can open the Use Image panel by *Option*-clicking any command that typically opens a Capture panel.

## File Browser

The upper portion of the Use Image panel is a standard file browser. Use it to navigate between the current suite's Images directory and subfolders.

## Command/Function Pop-Up Menu

To insert a command or function into the script with your image, choose it in the *Command/Function* pop-up menu.

To incorporate the image into the selected command or function in the script, choose *Add Image*.

## Max Wait Field

To specify the maximum wait time associated with the `Wait` or `WaitFor` commands, type the time (in seconds) in the *Max Wait* field.

## Cancel Button

To cancel the image selection and close the Use Image panel, click the *Cancel* button.

## Insert Button

The *Insert* button completes the Use Image operation. It inserts the generated command or function into the script, executes the command (if possible) and closes the Use Image panel.

## The Find Text Panel

---

In the Find Text panel, you can create text property lists— descriptions of text that you can use in place of captured images. To open the Find Text panel, click the *Find Text* button in the Viewer window, or choose *Control* menu > *Find Text*.

The Find Text Panel contains two sets of text properties; one for text platforms that use OCR (Optical Character Recognition), and one for platforms that dynamically generate images of your defined text.

## Generated Text Platforms

Text String: News

Platform: XP-TIG Style: Menu

Font: Tahoma

Size: 14 ☐ Bold

Text:  ☐ Italic

Background:  ☐ Underline

▼ Text Image Preview

News

Search Type: Text

Click Max Wait :

Insert into :ts.suite - CheckTest.script:  
Click (Text:"NewstStyle:"Menu", TextFont:"Tahoma")

Cancel Save

The Find Text Panel

### Text String Field

The *Text String* field contains the actual text that appears in your image. You must enter text here before you can insert the text property list.

### Platform Pop-up Menu

The *Platform* pop-up menu comprises the text platforms that you define in [Text preferences](#). Text platforms are usually distinguished by SUT operating system. Choose the Generic text platform or a text platform that represents the SUT you are working with. (For more information about text platforms and TIGs, see *Using EggPlant: "More About Text Platforms"*).

## Style Pop-Up Menu

The *Style* pop-up menu contains choices of predetermined text styles. After you choose a style as a starting point, you can change the individual attributes as needed. (For example, if your text uses a SUT's standard menu font but displays it in red, you can choose the menu style, then change the text color to red.)

You can define new styles in Text preferences.

## Font and Size Fields

In the *Font* and *Size* fields, choose the font and size of your text if they differ from the chosen style.

## Text and Background Color Wells

To edit a text or background color, click the respective color well and select a new color in the Colors panel.

**Tip:** The color picker in the Colors panel is especially useful for EggPlant; it allows you to copy a color from any place in your display, including the Viewer window. To use the color picker, click the magnifying glass in the Colors panel, then click again wherever you see the color you would like to copy.

## Bold, Italic and Underline Check Boxes

Select these checkboxes to generate bold, italicized, or underlined text in your text image. Not all fonts can be drawn in a bold or italic form, so some font and style combinations cannot produce a usable text image.

**Note:** Programs can differ in the way they draw underlining, so an underlined image may not produce the exact image you need. If this is the case, use *Capture Image* to take a snapshot of your text in the Viewer window.

## Text-Image Preview

To hide or show the image preview area, click the disclosure triangle or the words *Text Image Preview*.

Previews of text images can be useful if your selected text platform uses a native EggPlant text-image generator (TIG) or another fast TIG on the network.

If the selected text platform uses a slower TIG, or the platform's TIG is currently unavailable, the image preview can slow down your system.

## Search Type Pop-Up Menu

Search Type describes the level of precision EggPlant requires to consider a part of the Viewer window a match for your target image. For text, the best search type is usually Text. (For more information, see *Search Type* pop-up menu, in The Capture Panel, above.)

This pop-up menu is only available when the image preview area is shown.



## Command/Function Pop-Up Menu

To insert a command or function into the script with your text property list, choose from the *Command/Function* pop-up menu. The preview field below the pop-up menu shows the text that is inserted into your script.

To incorporate the text image into the selected command or function in the script, choose *Add Image*.

## Max Wait Field

To specify the maximum wait time associated with the `Wait` or `WaitFor` commands, type the time (in seconds) in the *Max Wait* field.

## Cancel Button

To cancel the text property list and close the Find Text Panel, click the *Cancel* button.

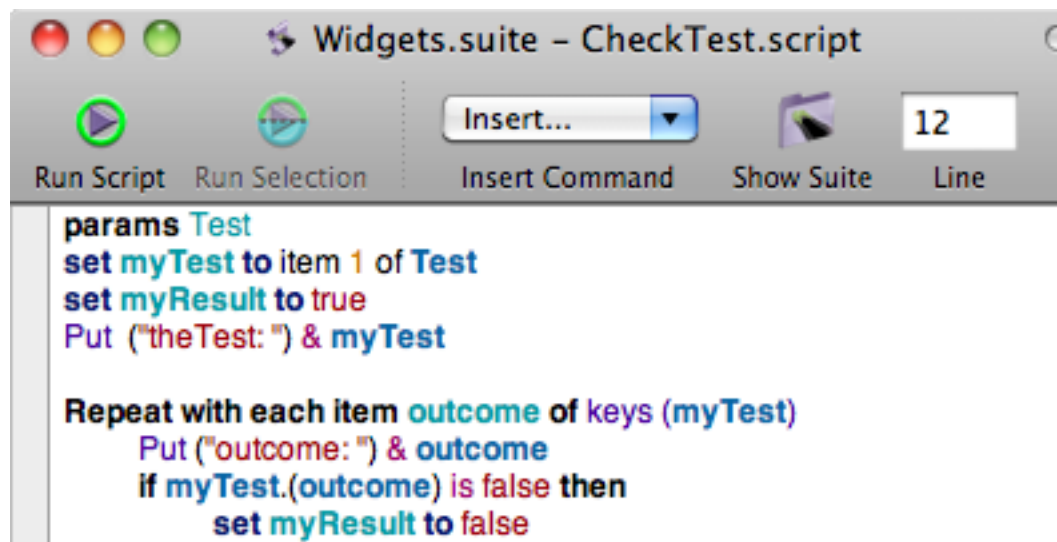
## Save Button

The *Save* button inserts the text property list into your script with the chosen command or function, executes the command (if applicable), and then closes the Find Text panel.

# The Script Editor

The Script Editor is the window in which you create and edit scripts.

To open the Script Editor, choose *File* menu > *New Script* or *Open Script*.



The Script Editor

## Creating and Editing Scripts

There are several ways to insert commands and images into a script:

- 1 Type commands, functions, and image names in the Script Editor.
- 2 Click the command buttons in the Viewer window toolbar.
- 3 Choose commands in the *Control* menu.
- 4 Choose commands in the Script Editor *Insert* pull-down menu. (See *Insert*, below.)
- 5 Copy, cut, and paste with the *Edit* menu or associated keyboard shortcuts.
- 6 Drag image, script, suite, and SUT names into the Script Editor from other windows.

## Auto-completion

When you have typed at least two characters in the Script Editor, *Complete* attempts to finish your current word. Possible completions include variables, commands, and names of scripts or images from your current suite.

To autocomplete a word, press F5 or Option-Escape, or choose *Edit* menu > *Complete*.

If there is a single word match, EggPlant completes the word for you. If more than one match is detected, choose the appropriate word and press Return, Tab, or Right Arrow.

To cancel the `Complete` command without selecting any of the choices, press `Escape`.

## Auto-indenting

Auto indenting puts tab or space characters inside control structures to make scripts easier to read. (Control structures include handlers, repeat loops, `if/then/else` and `try/catch` commands.)

By default, the Script Editor updates indentation when you press `Return` or `Tab`. (You can change this behavior in Script Editor preferences.)

## Colorization

By default, each type of script element (such as command, variable, and image name) is written with a different color in the Script Editor. This can make it easier to track what is happening in the script.

If you use incorrect syntax (that is, code that doesn't make sense to EggPlant) the entire line or block that contains the error is not colored; this is a visual cue that there is a problem in the script.

Colorization can be customized extensively in Script Editor preferences. (See [Script preferences in Licenses and Preferences](#).)

## menu features

The *Edit* and *Format* menus contain many of the features you find in word processors. Use these menus for standard editing tasks such as finding text, cutting and pasting, and changing font styles.

## Dynamic Breakpoints

Dynamic breakpoints are markers that cause a script to pause for debugging at designated points.

To set a dynamic breakpoint in the Script Editor, click in the column to the left of the script. A black triangle appears next to the line, denoting the breakpoint.

To change or remove a breakpoint, drag the triangle to another line in the script or out of the script entirely.

## The Script Editor Toolbar

---

The Script Editor toolbar buttons are described below. Since the toolbar can be customized, you might not see all of the buttons on your own toolbar. (For more information, see *Customize*, below.)

### Save

The *Save* button saves the current script. (When a script has unsaved changes there is a dot in the red close button in the top left corner of the window. The dot disappears when the script is saved.)

**Note:** By default, EggPlant automatically saves scripts when you run them. You can change this behavior in Script Editor preferences.

## Run Script / Abort Script

The *Run Script* button executes the script. While the script is running, the *Run Script* button is changed to *Abort Script*. (Clicking this button is the same as choosing *Run* menu > *Run Script* or *Abort Script*.)

While a script is running (and not paused), you cannot perform any manual actions in the Viewer window, or open and close VNC connections.

To run a script in debug mode, Option-click the *Run Script* button. This loads the script for execution and immediately pauses it, allowing you to step through line by line. (This is the same as choosing *Run* menu > *Debug Script*.)

## Run Selection

The *Run Selection* button executes only the selected part of the script. The selection does not have to include an entire line of code, but it must be syntactically complete and executable. (For example, if you select the beginning of an *if* block, you must select all the way through the *end if* statement.)

To run the selection in debug mode, Option-click the *Run Selection* button. This loads the selection for execution and immediately pauses it, allowing you to step through it line-by-line to closely follow its execution.

**Note:** When you run a selection, results are not generated in the Results pane of the Suite Editor.

## Line Jump Field

In this field, you can enter a line number and press Return to jump to that line in the script.

## Add Comment/ Comment/Uncomment



To insert a comment (*\*Comment\**) into a script, click the *Add Comment* button. The text of the comment is not compiled as part of the script.

When you select text in a script, the *Add Comment* button becomes *Comment*. To format the selected text as a comment, click the *Comment* button. (Because comments are not compiled with the rest of the script, you can use the comment button to temporarily disable parts of a script.)

When you select a comment in the script, the *Add Comment* button becomes *Uncomment*. To remove the comment formatting the selected text, click the *Uncomment* button. The selected text is then treated as a regular part of the script.

## Insert Pull-Down Menu

The *Insert* pull-down menu contains image-related commands and functions. When you choose one of the items from this menu, an *Open File* dialog allows you to select previously-captured images as parameters to the chosen command or function. (Choose *Additional Image* to insert an image into your script with no new command or function.)

To insert a text property list into the script, shift-click the *Insert* pull-down menu.

**Note:** You can also open this menu by right-clicking in the Script Editor.

## Handler Pop-Up Menu

The *Handler* pop-up menu contains a list of all of the handlers defined in the current script. You can choose a handler to jump to its definition in the script.

## Show Suite

The *Show Suite* button opens the script's suite in a Suite Editor.

## Show Resource

The *Show Resource* button opens a script or image named in the currently selected text.

**Note:** You can also show a resource by Option-clicking a script or image name.

## Show Results

Click the *Show Results* button to open the last recorded results for this script in the Results pane of the Suite Editor window.

## Customize

The *Customize* button brings up a panel in which you can select the buttons that are available in the Script Editor toolbar.

To add a button to the toolbar, drag the button from the Customize panel to the toolbar;  
To remove a button from the toolbar, drag it off of the toolbar.

While the Customize panel is open, you can also drag buttons to different locations within the toolbar.

## Add Image

The *Add Image* menu item inserts the image into the script with no new command or function.

## Find Text

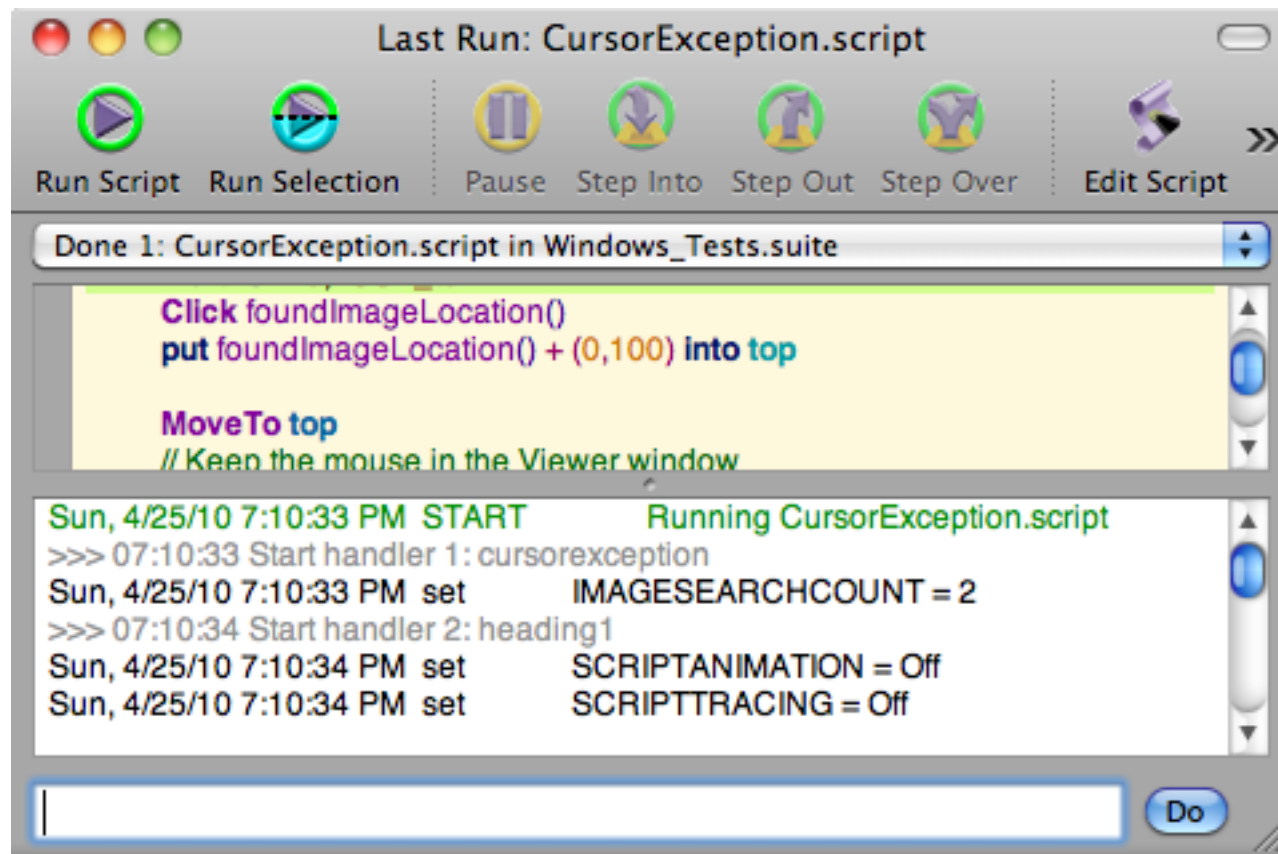
The *Find Text* button opens the Find Text panel, which allows you to create an image based on text attributes. (Shift-clicking the *Insert* pull-down menu also opens the Find Text panel.) (For more information, see [The Find Text Panel.](#))

## Colors, Fonts, and Print

These buttons open the standard color selection, font, and print panels found in many applications.

## The Run Window

The Run window displays information about the currently running script. In the Run window you can manually control script execution, view script output as it is generated, and follow the progress of the script with animation or tracing.



The Run window

## Script Display area

The Script Display area is the top half of the Run window. When script animation is turned on, the currently executed handler is displayed here. (When script animation is turned off, the initial handler is displayed, but it is only updated when the script is paused.)

**Note:** A handler is a unit of execution in SenseTalk. If a script does not explicitly declare a handler, the entire script is treated as a handler. (For more information on handlers, see the *SenseTalk Reference Manual*.)

## Dynamic breakpoints

Dynamic breakpoints are markers that cause a script to pause for debugging at designated points. They can be added or changed at any time, even while the script is running.

You can set a dynamic breakpoint in the Run window by clicking in the column to the Script Display area. A black triangle appears next to the line, denoting the breakpoint. (You can also set a breakpoint at your insertion point in the

script by choosing *Run menu > Set Breakpoint.*)

You can move a breakpoint by dragging the triangle to another line in the script, or remove it by dragging it out of the column.

## Script frame Pop-up Menu

The *Script Frame* pop-up menu displays the handlers that are being executed, listed in the reverse order in which they are called.

When script execution is paused, you can select a different frame from this pop-up menu to view it in the Script Display area and see which line it is currently executing. This also sets the context for commands typed into the Ad-hoc Do Box. For example, if you want to output the value of a variable that is present only in the script handler that started your execution, you must make sure to select the very bottom frame, numbered “1.”

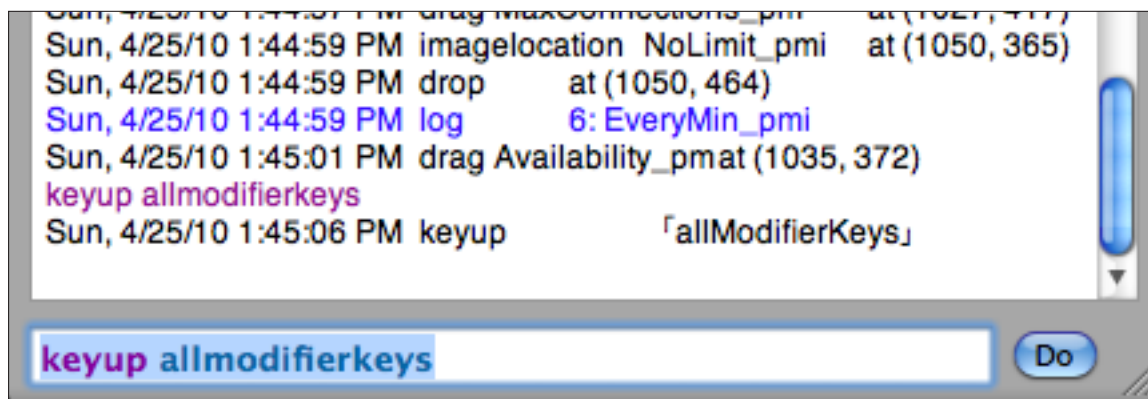
## The Log Area

The Log Area is the lower half of the Run window. As the script runs, the Log Area displays the following information:

- 1 The same information that is recorded in the script’s permanent log. (See the Results pane, in The Suite Editor.)
- 2 The output of **Put** commands that have no specific destination.
- 3 Script tracing information, when Script Tracing is turned on. (See [the ScriptTracing](#), in *Global Properties.*)
- 4 Output from commands entered in the Ad-hoc Do-Box. (See the Ad-hoc Do-Box, below.)

## The Ad-hoc do box

The Ad-hoc Do Box (AHDB) allows you to modify a script as it is running. For example, you can assign new values to variables, execute commands, or call other scripts.



*The Ad-hoc Do Box*

## The Run Window Toolbar

---

The Run window toolbar buttons are described below. Since the toolbar can be customized, you might not see all of the buttons on your own toolbar. (For more information on adding and removing toolbar buttons, see *Customize* below.)

### Run Script/Abort Script

The *Run Script* button reruns the last script that was run, or the entire script from which a selection was run. While the script is running, the *Run Script* button becomes *Abort Script*.

While a script is running (and not paused), you cannot perform any manual actions in the Viewer window, or open and close VNC connections.

**Note:** You can run the script in debug mode by Option-clicking the *Run Script* button. This loads the script for execution and immediately pauses it, allowing you to step through line by line. (This is the same as choosing *Run* menu > *Debug Script*.)

### Run Selection

The *Run Selection* button executes only the selected part of a script. The selection does not have to include an entire line of code, but it must be syntactically complete and executable. (For example, if you select the beginning of an **if** block, you must select all the way through the **end if** statement.)

You can run the selection in debug mode by Option-clicking the *Run Selection* button. This loads the selection for execution, but immediately pauses it, allowing you to step through it line-by-line to closely follow its execution.

**Note:** When you run a selection, results are not generated in the Results pane of the Suite Editor.

### Pause / Continue

The *Pause/Continue* buttons allow you to pause and continue execution whenever a script is running.

When the script execution is paused, you can perform additional operations:

- 1 Type commands in the Ad-hoc Do-Box. (See *The Ad-hoc Do-Box*, above.)
- 2 Use the debug mode tools: *Step Into*, *Step Over*, and *Step Out*. (See below.)
- 3 Interact with the Viewer window in Live Mode.



## Step Into

The *Step Into* button executes the next line of a script, then pauses. If this line calls another script or handler, the script “steps into” the first line of the called script or handler before pausing again.

**Note:** The *Step Over* button performs a similar function, with one exception: If the line calls another script or handler, *Step Over* executes the called script or handler entirely before pausing.

## Step Over

Like *Step Into*, *Step Over* executes the next line of the script, then pauses. However, if this line calls another script or handler, that script or handler runs in its entirety, “stepping over” it to the next line of original script before pausing again.

During the execution of the script or handler that is stepped over, you can click *Pause* to regain control of the execution before the *Step Over* command has finished.

## Step Out

Step Out executes all the remaining lines in the current script or handler, then pauses. (Using Step Out on your initial handler is the same as clicking Continue.)

During the execution of the code being stepped out of, you can click *Pause* to regain control before the Step Out command has finished.

## Animation Pop-Up Menu

Animation highlights the current line of code in the Script-Display area of the Run window as the script is being run. You can change animation settings at any time with immediate effect, including while a script is running.

Animations cause scripts to execute more slowly, because EggPlant is doing more work at the display level. You can improve script execution speed by turning off all animations when you are not debugging.

## Animation Off

When you choose Animation Off, the Display area of the *Run* window does not update as the script runs; only the initial handler for the script is shown, unless the execution is paused.

## Animate Calls

When you choose *Animate Calls*, the *Run* window shows each script or handler as it is called, but it does not high-

light each line of the script.

## Animate All

When you choose *Animate All*, the Run window highlights each line of each script or handler as it is run.

**Note:** Animation settings can also be modified temporarily by scripts. (For more information, see [the ScriptAnimation in Global Properties.](#))

## Tracing Pop-Up Menu

Tracing causes EggPlant to “echo” information about what the script is about to execute to the Log Area in the Run window. This is similar to animation, but it allows you to review *all* of the code that is executed during the script run, interspersed with logged output. This can provide more context when debugging a script.

You can change tracing settings at any time with immediate effect, including while a script is running.

**Note:** Tracing settings can also be modified temporarily by scripts. (For more information, see [the ScriptTracing in Global Properties.](#))

## Tracing Off

When you choose *Tracing Off*, only standard log messages are displayed in the Log Area of the Run window.

## Trace Calls

When you choose *Trace Calls*, the Log Area of the Run window displays a message as it enters and leaves each script or handler.

## Trace All

When you choose *Trace All*, the Log Area of the Run window displays every line of each script or handler prior to executing it.

**Note:** Tracing cause scripts to execute more slowly, because EggPlant is doing more work at the display level. You can improve script execution speed by turning off tracing when you are not debugging.

## Edit Script

The *Edit Script* button opens the Script Editor for the script that is currently in the Run window. If you have text selected when you click this button, EggPlant attempts to automatically select the same text in the Script Editor. (If the selected text has been edited since the last run, EggPlant may not be able to find it in the Script Editor.)

## Show Suite

The *Show Suite* button opens the current script's suite in the Suite Editor.

## Show Results

The *Show Results* button opens the last recorded results for the current script, in the Results pane of the Suite Editor.

## Customize

The *Customize* button brings up a panel in which you can select the buttons that are available in the Run window toolbar.

To add a button to the toolbar, drag the button from the Customize panel to the toolbar;

To remove a button from the toolbar, drag it off of the toolbar.

While the Customize panel is open, you can also drag buttons to different locations within the toolbar.

## The Suite Editor

The Suite Editor is the window in which you manage scripts, images, and logs; it is the control hub for all of your testing operations.

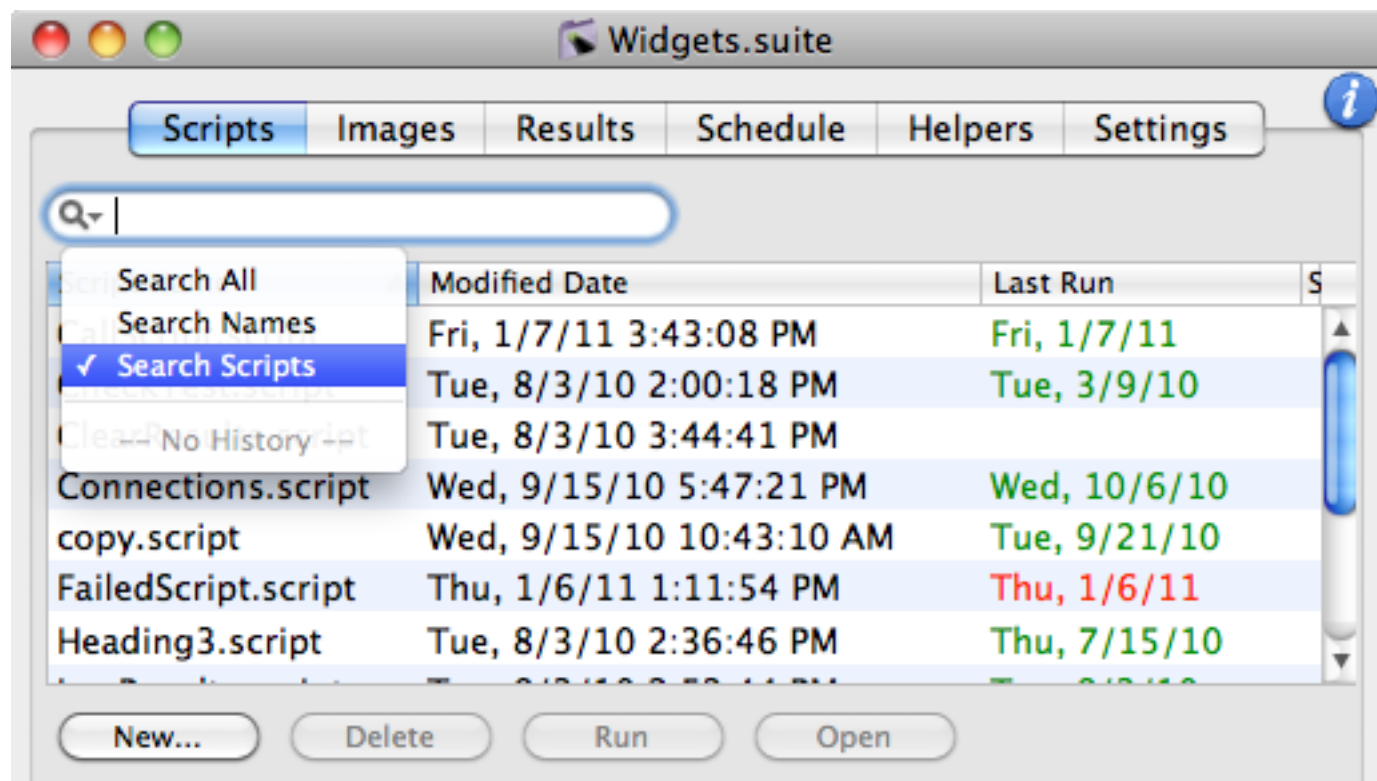
When you are working with multiple suites, you must open a separate Suite Editor for each one.

To open a Suite Editor, choose *File menu > New Suite or Open Suite*.

Note: If a suite folder is locked in your file system, a lock icon appears in the top left corner of the Suite Editor. When a suite is locked, you cannot add, remove, or edit any script resources. You can still run scripts, but run data is not saved.

## The Scripts pane

The Scripts pane contains a list of all of the scripts in the suite. For each script, the Script Name, Modified Date, Last Run Date, and Size are displayed.



The Scripts pane

## Searching for a Script

To search for a particular script in your suite, type part of a script name or script text in the search field above the script list. To set the type of search filter, click the magnifying glass in the *Search* field, and choose *Search All*, *Search Names*, or *Search Scripts* from the pop-up menu.

## Creating a new script

To create a new script, click the New button. The script is added to the Scripts pane, and a new Script Editor opens.

## Deleting scripts

To delete the selected script (or scripts), click the Delete button or drag the scripts into the Trash. All Script Editors associated with deleted scripts close.

## Opening scripts

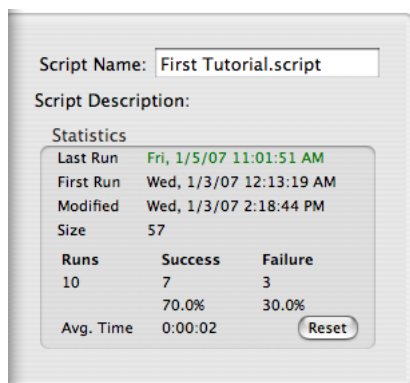
To open the selected script (or scripts), click the Open button, or double-click a script name.

## Running a script

To run the selected script, click the Run button, or choose Run menu > Run Script.

## Info

The Info button opens and closes a panel that contains supplemental information about the selected script:



*The Script info panel*

**Script Name:** This text field contains the name of the script. If you edit the name, it is also changed in the Results pane. (The underlying Results folder also reflects the name change.)

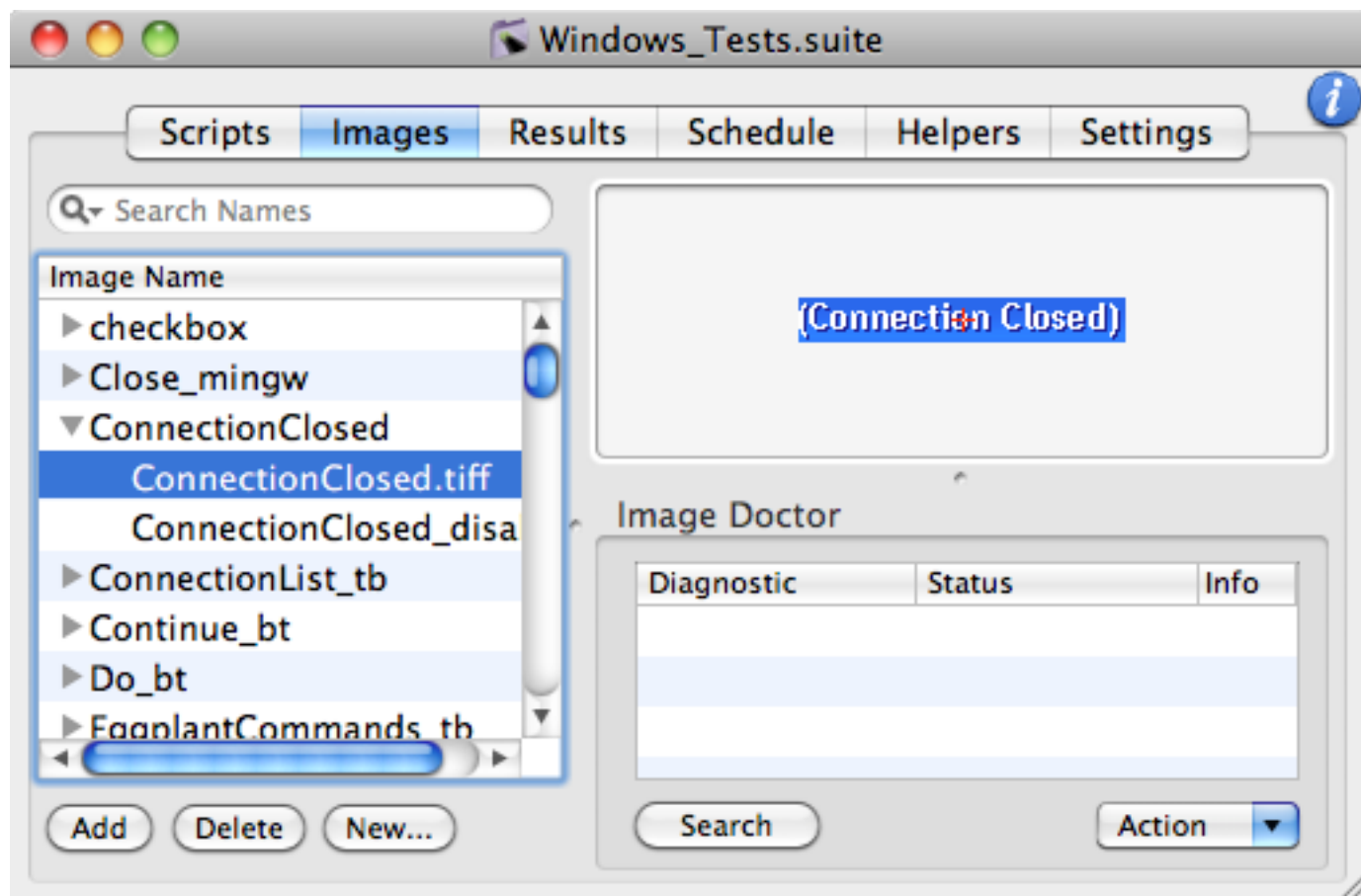
**Note:** If you change the name of a script, remember to edit calls to that script to reflect the new name.

**Script Description:** This text field is a good place to record the purpose of the script, or other pertinent notes.

**Statistics:** The bottom of the info panel displays the script's run history. You can reset the Statistics area by clicking the Reset button.

## The Images pane

The Images pane contains a list of all of the image subfolders and images saved in the suite. (Image subfolders are displayed first, followed by the images they contain.)



The Images pane

## Adding Images and Image Folders

There are two ways to add outside images to an EggPlant suite:

- 1 Drag one or more images or image folders from another location into the image list in the Suite Editor.
- 2 Click the *Add* button below the image list, and select one or more images or image folders in the file browser..

**Note:** EggPlant accepts the following image file formats: TIFF, ICNS, ICO, PICT, GIF, BMP, PNG, PDF, and JPG / JPEG.

## Searching for Images

To search for images in your suite, type an image name or part of an image description in the *Search* field above the image list. To set the type of search filter, click the magnifying glass in the *Search* field, and choose *Names*, *Descriptions*, or *All* from the pop-up menu.

**Tip:** To save your search term as an item in the *Search* pop-up menu, press *Return* after you type the term.

## Deleting Images and Image Folders

There are two ways to delete images and image folders from an EggPlant suite:

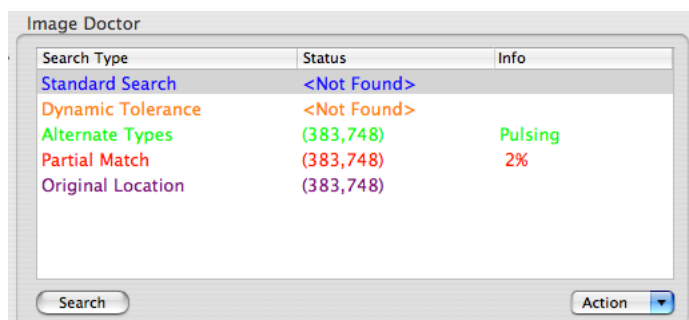
- 1 Select one or more images or image folders in the image list, and click the *Delete* button.
- 2 Drag one or more images or image folders from the image list to the Trash.

## Creating New Image Folders

To create a new image folder, click the *New Folder* button below the image list and enter a folder name.

## The Image Doctor

The Image Doctor is a utility that adjusts the search criteria for images that are not being found in the Viewer window. To use the Image Doctor, select an image in the image list, and click the Image Doctor *Search* button.



*The Image Doctor*

## Diagnostic

The labels in this column describe each kind of adjustment the Image Doctor makes. (For more information on search criteria, see [Image property list in EggPlant Commands and Functions](#).)

- **Standard Search.** This is the normal state of the image with no adjustments.
- **Dynamic Tolerance.** This version of the image starts at the image's normal tolerance setting and gradually raises the tolerance until the image is found in the Viewer window.
- **Alternate Types.** This version applies increasingly lenient search types until a match is found.
- **Discrepancy.** This version gradually increases the image's discrepancy setting until a match is found.

## Status

This column displays the *found* status of each diagnostic. If the diagnostic is found in the Viewer window, the screen coordinates of the image are displayed in the *Status* column.

## Info

This column displays information about how much each successful diagnostic changes the search criteria to produce

a match.

## Using the Diagnostics

The Action drop-down menu contains several ways to use a selected diagnostic:

- **Show Original Image.** This shows the diagnostic image with your original image overlaid to help you discern the difference between the two.
- **Fix Image.** This replaces your images original search criteria with the search criteria that produced the diagnostic image match.
- **Recapture Image.** This recaptures your image at the location where the diagnostic image is found.
- **Add Representation.** This turns your original image into an Image Collection, and saves the diagnostic image as part of it. (For more information, see [Image Collection in EggPlant commands and functions.](#))
- **Edit Script.** If you are using the Image Doctor as a result of a script error, *Edit Script* opens the Script Editor and selects the line that caused the error.
- **Continue Script.** If your script run is paused because of a script error, *Continue Script* resumes the run.

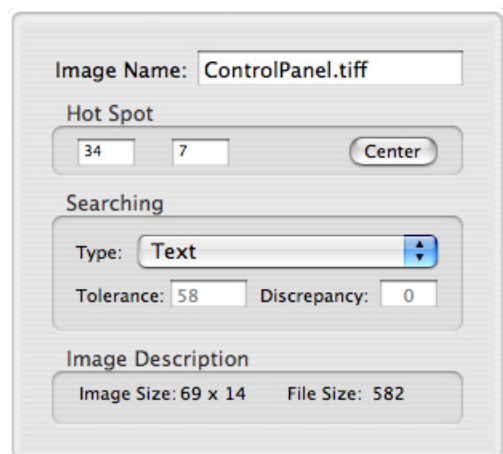
## Thumbnail view

The top right section of the *Images* pane shows the selected image, scaled to fit as necessary.

The image's Hot Spot is marked by red cross hairs; you can change it by Command-clicking in a new location. You can also nudge the Hot Spot one pixel at a time with Command-Arrow key. (Adding the Shift key moves the Hot Spot in ten-pixel increments.)

## Info

The *Info* button opens and closes a panel that contains the following supplemental information about the selected image:



The screenshot shows a light gray panel titled 'Image info panel'. It contains several sections: 'Image Name' with a text field containing 'ControlPanel.tiff'; 'Hot Spot' with two numeric input fields (34 and 7) and a 'Center' button; 'Searching' with a 'Type' dropdown menu set to 'Text', and 'Tolerance' (58) and 'Discrepancy' (0) input fields; and 'Image Description' with 'Image Size: 69 x 14' and 'File Size: 582'.

*The Image info panel*



- **Image Name.** This text field contains the name of the image.
- **Hot Spot.** The values in these fields represent the horizontal and vertical distance from the top left corner of the image to the Hot Spot, measured in pixels. You can move the Hot Spot by editing these values, or Command-clicking in the thumbnail view. (The *Center* button moves the Hot Spot to the center of the image.)
- **Search Type.** This pop-up menu contains the settings that determine how precisely the Viewer window must match the image for a successful result. (For more information, see [Search Type](#).)
- **Tolerance.** The number in this field represents the acceptable difference between RGB color values in the image and a match in the Viewer window. (If you enter a single value, it is used for all three color channels. You can also enter three separate values, separated by commas.)

**Note:** When you edit the Image Tolerance value, it is no longer automatically changed when you choose a different search type. If you delete this value, the image is effected by search type again. [The Screen Tab, in Run Option preferences](#).)

- **Discrepancy.** A percentage in this field (including the percent sign) indicates the percentage of pixels that may differ between the actual image and a match in the Viewer window. A number in this field (with no percent sign) indicates the absolute number of pixels that may differ between this image and a match in the Viewer window.
- **Image Description.** This area displays the image size in pixels, and the image file size in bytes. You can type comments about the image in the text field, and use the text of these comments as image search terms, or CollectionFilter values. (For more information, see [Searching for Images](#) above, or [the CollectionFilter in Global Properties](#).)

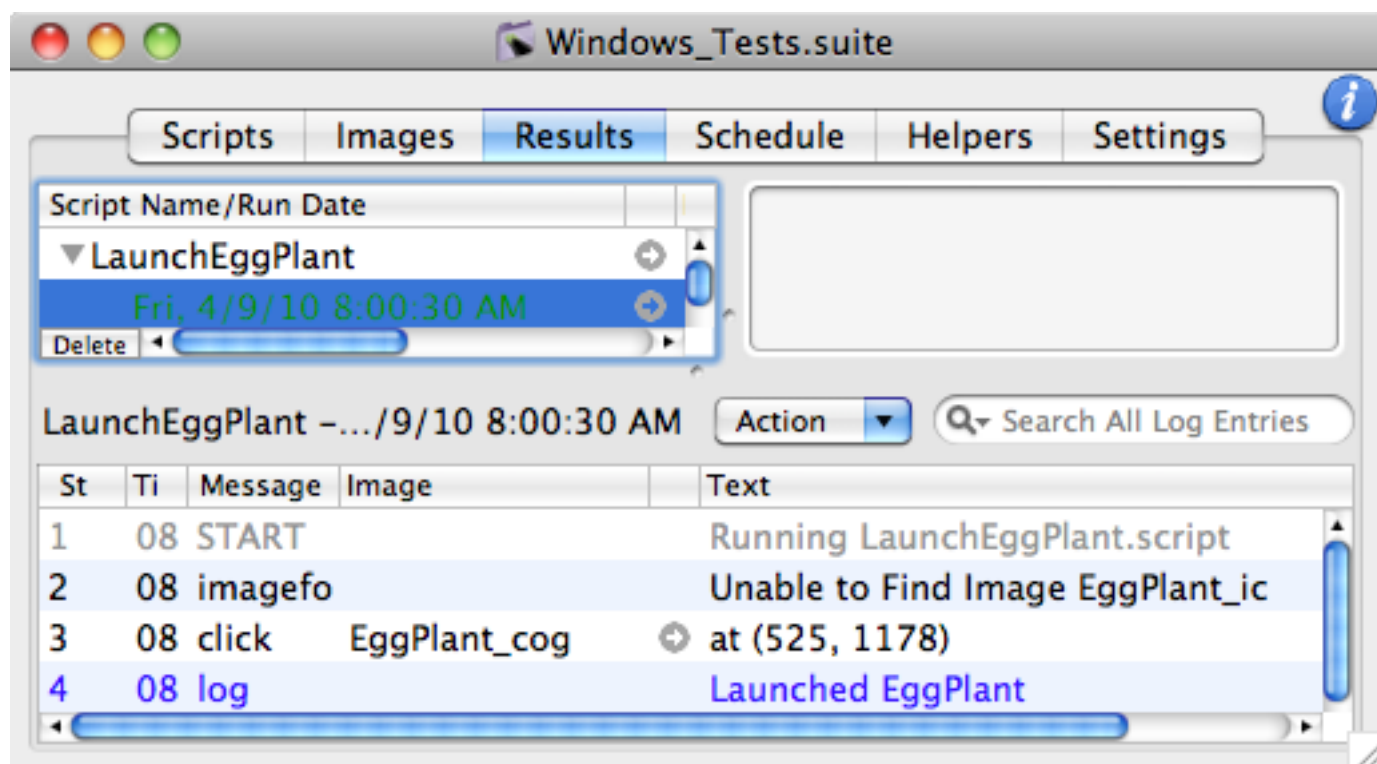
## Deleting Images or Image Folders

To delete the selected images and/or folders, drag them to the Trash, or click the Delete button.

## The Results pane

---

The Results pane contains records of the tests you have run in the current suite. For each script each test you've run for each script by date and time; the complete log for each test; and images for any step in the log including full screen images captured in cases of script failure.



The Results pane

## The Script Name/Run Date List

Every script that you test is recorded in the Script Name list; you can click a script name to show or hide the dates and times when that script was tested.

Script dates are color-coded to denote test results:

- **Green.** Success
- **Red.** Failure
- **Black.** Aborted or Interrupted

For each Run Date, the list displays the number of errors, warnings, and exceptions that occurred in the run. (For more information, see [The ScriptLogging](#), in [Global Properties](#).)

## Resource Links

Next to each script name and run date, there is an arrow that links a results resource. For scripts, the arrow opens a .CSV text file that contains overall statistics for the script. For run dates, the arrow opens the LogFile.txt file, which contains a line-by-line record of the run.

## Deleting Script Names and Run Dates

The Delete button deletes the selected Script name or Run Date and associated results resources. This action requires confirmation.

Run Date deletions do not change a script's statistics.

## The Log Area

When you select a Run Date in the list, the Log Area shows the detailed log for that run.

The log shows each step of the script that executed an EggPlant command or function; you can double-click any step to open the Script Editor with that step highlighted.

**Note:** The EggPlant commands are an extension of the SenseTalk scripting language, and they are described in this manual. (For more information on using the core Sensetalk language for advanced scripting, see the *SenseTalk Reference Manual*.)

For each numbered line of the tested script, the Log Area provides the following information:

### Time

The Time column shows the local system time when each step occurred. Times are recorded to the nearest thousandth of a second. (You might have to widen the column to see fractional seconds.)

### Message

The Message column displays the name of the function or command that was executed. (In SenseTalk, command and function calls are referred to as *messages*.)

If an error occurred in the line, Exception is displayed. If the exception was not caught, the following line displays *Failure*.

### Image

The Image column displays the name of any image used by the command or function in that step. (For a text property list, the word *Text* is shown, followed by the text of the image.)

If the step called the `CaptureScreen` command, the name of the captured image is displayed.

If the step is the last step of a failed script, `Screen_Error` file is displayed.

### Resource link

When the Image column displays the name of an image (or movie on Mac OS X), the Resource Link column contains a link to the related resource, as shown in the table below.

## Log Area resource links

Image column value	Resource opens with...
Captured Image	Image pane of the Suite Editor
Generated text property list	Your default image-viewing program
Screenshot	Your default image-viewing program
Movie (currently Mac OS X only)	QuickTime

## Text

The Text column displays the additional information shown in the table below.

## Additional log information in the text column

Message or image value	Associated text
TypeText	Keys that were typed in the Viewer window
Image	Coordinates where image was found
Error	Text of the exception that was thrown
Command that changes option settings	New setting values

## Colorization

Some steps in the Log Area are colorized to indicate results. The meaning of each color is shown in the following table.

## Log Area colorization

Color	Meaning
Blue	Log command executed
Orange	LogWarning
Red	LogError
Green (displayed in the last line)	Successful Test
Red (displayed in the last line)	Failed Test

## Searching for Log Entries

To search for log entries, type a search term (such as a command or image name) in the *Search* field above the Log Area. To set the type of search filter, click the magnifying glass in the *Search* field, and choose *All Entries*, *Interesting*

*Entries*, or *Warnings & Errors* from the pop-up menu. (Interesting Entries include: warnings, errors, exceptions, log commands, start, and end.)

**Tip:** To save your search term as an item in the Search pop-up menu, press *Return* after you type the term.

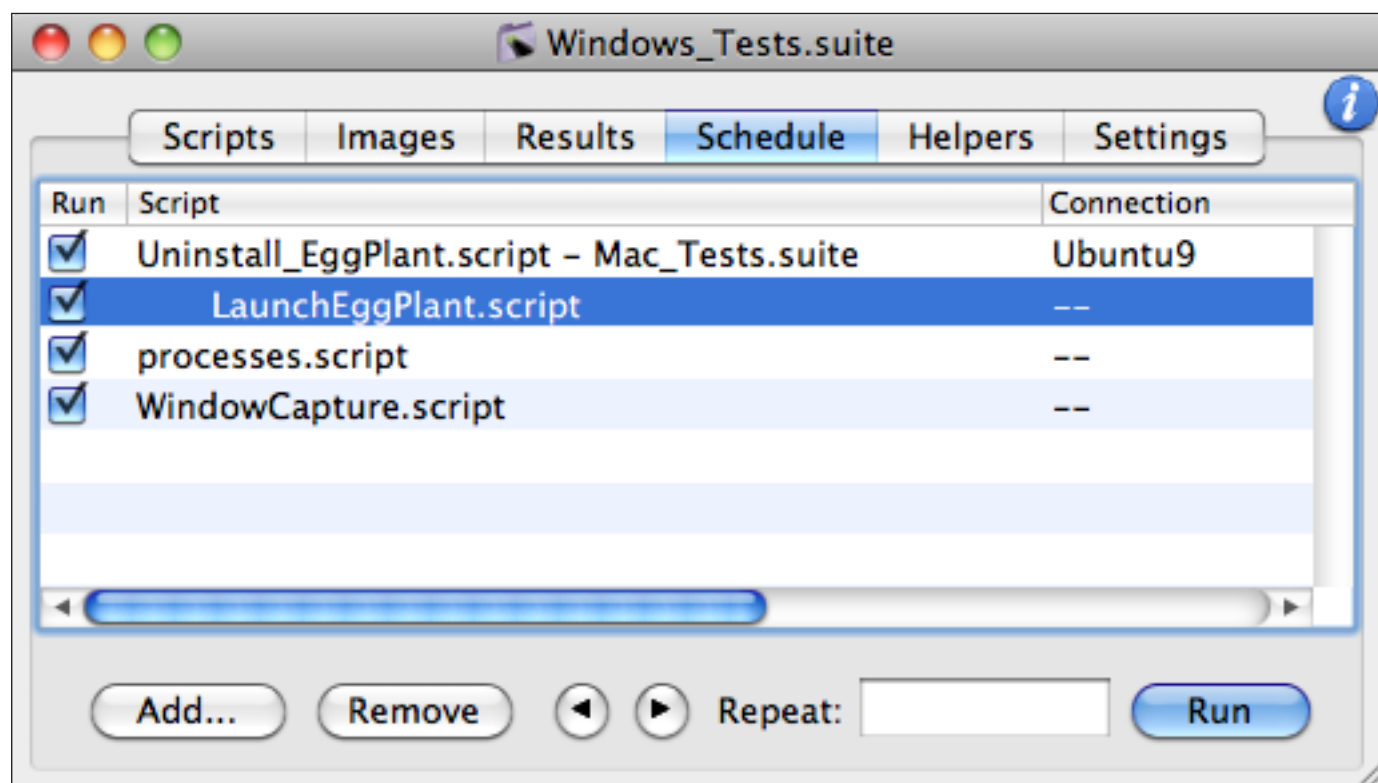
## Action Drop-Down Menu

The *Action* drop-down menu, above the Log Area, contains several actions that you can perform on the selected Log entry in the Log Area:

- **Show Script Line:** Opens the Script Editor and highlights the line of the script associated with the selected Log entry.
- **Show Image:** Switches to the *Images* pane, and shows the image associated with the selected Log entry.
- **Apply Fix:** For log entries that were generated by the Auto Doctor, this action permanently applies an Auto Doctor fix to the selected image.

## The Schedule pane

The Schedule pane contains a list of current and upcoming script runs. (The scripts can come from any suite, not just the active one.)



The Schedule pane

## Adding Scripts to the Schedule

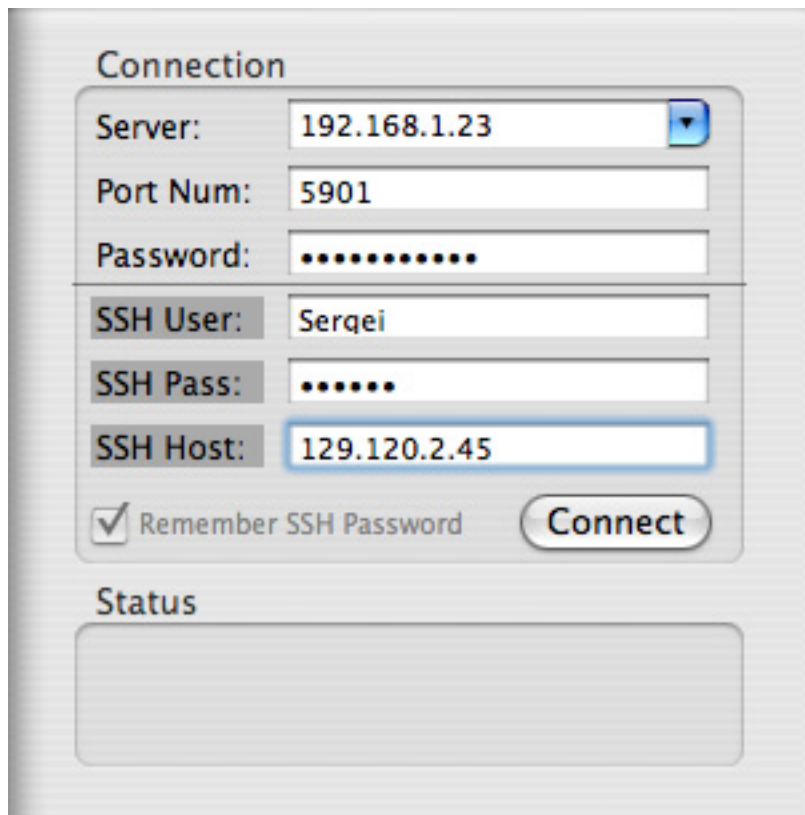
There are three ways to add scripts to the schedule:

- 1 Click the Add button on the bottom of the Schedules pane; select the script from the Add panel.
- 2 In the Scripts pane, drag script names to the Schedules pane.
- 3 In the Script Editor title bar, drag the script icon to the Schedules pane in the Suite Editor.

## Providing a Script's Connection Details

When you run a script from the Schedules pane, you can provide connection information to specify a SUT for the script run.

To specify a SUT for a selected script, click the Info button to open the Info panel. There you can enter a host name or IP address, VNC port number, and password. (If you need to establish a secure connection, you can also provide connection and login information for an SSH host.)



The screenshot shows a dialog box titled "Connection" with several input fields and a "Connect" button. Below the "Connection" section is a "Status" section with a large empty text area.

Connection	
Server:	192.168.1.23
Port Num:	5901
Password:	.....
SSH User:	Sergei
SSH Pass:	.....
SSH Host:	129.120.2.45
<input checked="" type="checkbox"/> Remember SSH Password	
<button>Connect</button>	

**Status**

*The Schedule info panel*

## Changing the Run Order

You can move a script to a different place in the schedule by dragging it.

To move a range of scripts, Shift-drag the script names to a new location.

To move multiple nonsequential scripts together, Command-click and drag the script names.

## Duplicating Scripts in a Schedule

To schedule additional runs of a script in the schedule, Option-drag the script name as if to move it. (You can even add the Option key when you are already dragging a script name.) The original script remains, and a copy is created wherever you drop the script name.

To duplicate a range of scripts together, Option-Shift-drag the script names.

To duplicate multiple, nonsequential scripts, Option-Command-click and drag the script names.

## Setting Dependencies

In the Schedules pane, you can schedule a script to run only if the preceding script is successful.

To set up a dependency, select one or more scripts that depend on the success of the previous script, and click the right arrow button or press the Right-Arrow key. The script is indented to show the dependency. (Click the left arrow button or press the Left-Arrow key to remove the dependency.)

You can create many levels of dependency (a domino effect) by indenting multiple times.

Run	Script
<input checked="" type="checkbox"/>	Pamela.script
<input checked="" type="checkbox"/>	Untitled.script
<input checked="" type="checkbox"/>	script1.script
<input checked="" type="checkbox"/>	script1.script

*Script dependencies*

**Tip:** Although the Schedules pane does not support dependencies based on script failures, you can accomplish this with a master control script. (For more information, see "Writing Master Scripts", in the *Using EggPlant* manual.)

## Disabling and Removing Scripts from the Schedule

To temporarily disable a script in the schedule, deselect its checkbox in the Run column. (This also disables the script's dependent scripts.)

To permanently remove a selected script from the schedule, click the *Remove* button.

## Running the Schedule Manually

To run the scripts in a scheduled list, click the *Run* button. If you provide connection information for a script, EggPlant opens a VNC connection with the specified SUT before running the script. If you do not provide connection information, the script is run on the currently active SUT. (If there is no connection information and no current VNC connection, the script fails.)

To run the entire batch of scripts more than once, enter the appropriate number in the *Repeat* field before you click

the *Run* button.

**Note:** If a script fails when it is run from the Schedules pane, the failure is noted in the *Status* column of the Script List, and EggPlant moves on to the next script; there is no command or error that prevents a schedule from running to completion.

## Running the Schedule at a Particular Time

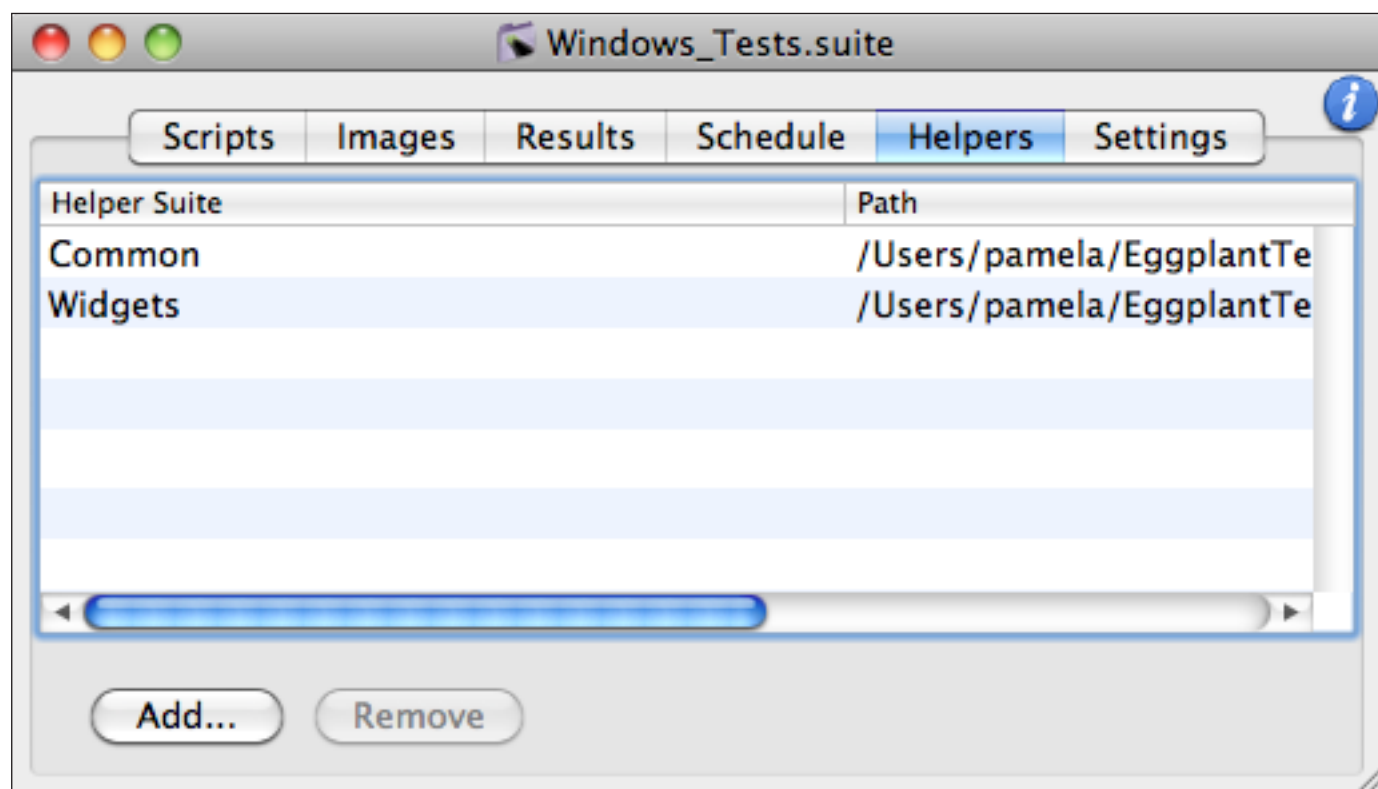
Here are two ways to run the script schedule at a designated time:

- 1 Use the Unix `crontab` or `at` command.
- 2 Include the command-line call in a build script or some other shell script that is already running automatically. (This is often done to verify that a new build of an application does not have any defects before it is delivered to QA or other parts of the organization.)

## The Helpers pane

The Helpers pane contains a list of the suites whose scripts, images, and helpers are available to the current suite (helper suites.) Helper suites are useful as core suites, with scripts and images that you use across several other suites.

The file path of each helper is displayed in the Path column; red text indicates a file path that is no longer valid.



*The Helpers pane*



## Adding a Helper Suite

There are two ways to add a helper in the Helpers pane:

- Click the Add button, and select a suite in the Add panel.
- Drag the suite icon from the title bar of a Suite Editor to another suite's Helpers tab.

## Using a Relative Path for a Helper Suite

You can change the path of a helper suite to make it relative to your Home directory or your Default Suite Directory. To start editing the path of the helper suite, select the helper suite in your Helpers pane and click in the path.

Paths that start with tilde (~/) are relative to your Home directory. Paths that start with a dot (./) are relative to your [Default Suite Directory](#).

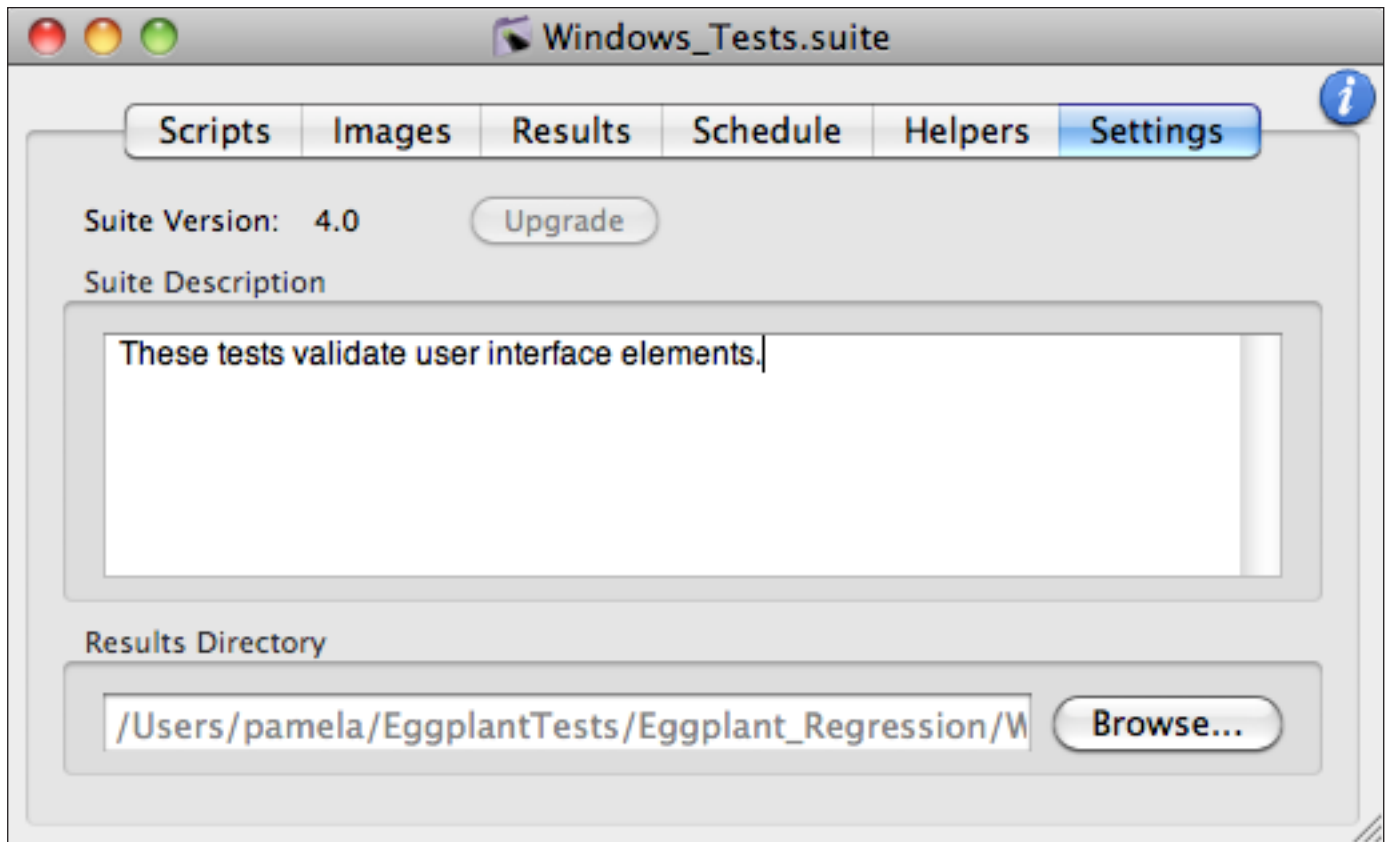
## Removing a Helper Suite

To remove a helper suite, select it and click the Remove button. When you remove a helper suite, scripts in the current suite can no longer access that helper suite's scripts, images, and other helper suites.

## The Settings pane

---

The Settings pane contains general information about the suite, including your notes and the location of the Results directory.



*The Settings pane*

## Suite Version

Displays your EggPlant version number. To automatically update your scripts that are effected by changes to the SenseTalk scripting language, click the *Update* button.

## Suite Description

This text field is a place for you to record your own notes about the current suite.

## Results Directory

This text field contains the file path of the current suite. Click the *Browse* button to select a new directory in the file browser.

## Licenses and Preferences

### The License Registry panel

---

The License Registry panel displays information about the EggPlant Licenses available and in use on your network. When you launch EggPlant, the license registry checks to make sure there is a license available.

### Entering a New License

Fill in the *Key* field and *User Name* field.

EggPlant finds your *HostID* in your computer's System preferences.

### Removing a License

Select the license in the *Licenses Found* list, and click *Remove*.

**Note:** You can not remove a license that is currently in use.

### Requesting a Free Trial

Click the *Free Trial* button to request a trial EggPlant license.

## General preferences

---

The General preferences control several basic “housekeeping” behaviors.

### On Startup

- **Reopen suites and scripts.** Select this checkbox to automatically reopen your last open suites and scripts when you launch EggPlant.
- **Restore previously-open connections.** Select this checkbox to automatically reopen your last open VNC connections when you launch EggPlant. (If EggPlant cannot connect to a previously-open SUT, there is no alert or error message.)

### Connection List

- **Update connection availability.** This pop-up menu contains choices of how often SUT status in the Connection List is updated. (Regardless of your choice, SUT status is always updated when you first open or close a VNC connection.)
- **Maximum # of open connections.** This pop-up menu allows you to limit concurrent SUT connections to the chosen number.
- **Close Connection List upon opening a connection.** When this checkbox is selected, EggPlant closes the

Connection List whenever you open a VNC connection.

- **Show alert when a connection fails.** When this checkbox is selected, EggPlant prompts you to retry or cancel when a VNC connection fails.
- **Enable Bonjour discovery of VNC servers.** When this checkbox is selected, local computers that have Bonjour networking technology enabled automatically appear in your Connection List.

**Note:** SUTs that are detected through Bonjour are not automatically saved to the Connection List. To save a Bonjour-detected SUT, edit its connection information; otherwise, the SUT is removed from the Connection List when it is no longer detected on the network.

## Other Options

- **Automatically upgrade suites to the current version.** When this checkbox is selected, EggPlant automatically changes any of your scripts that are effected by changes in SenseTalk commands and functions.
- **Bring the Run window forward when a script is run.** Select this checkbox to automatically open the Run window (or bring it to front) whenever a script is run.
- **Alert user when script execution fails.** Select this checkbox to see an alert dialog when a script fails. (This does not apply to scripts that are run as part of a Suite Editor schedule.)
- **Store thumbnail icons for captured images.** Select this checkbox to generate preview images in file system views and preview areas. (Deselect this checkbox if you are using a file system or version control system that does not support resource forks.)

## Default Suite Directory

This field contains the default location EggPlant presents when you open a New Suite, Save As, or Open dialog. Type a file path in the text field, or click the Browse button and navigate to the directory. This is also used as the initial setting of the Folder global property at the beginning of each script run.

## Viewer Window preferences

---

The Viewer window preferences control the ways you can interact with the SUT through the Viewer window.

### Live Mode Toggle Key Pop-Up Menu

Choose the key used for quick toggling between Capture Mode and Live Mode. (The default choice is Command.)

### Live Mode Cursor Pop-Up Menu

This setting determines the appearance of your cursor in the Viewer window. Choose one of the following in the pop-up menu:

- **Local Cursor.** This option displays the local EggPlant cursor in addition to any visible SUT cursor.
- **Guide Box.** This option displays the local cursor as a small guide box, in addition to any visible SUT cursor.
- **None.** This option displays only the cursor that is native to the SUT, and displayed by its VNC server.

## Mouse Scroll Wheel Pop-Up Menu

This setting determines whether mouse scrolling in a Viewer window is interpreted by the local Viewer window, or by the active window in the Viewer window. Choose one of the following:

- **Remote System.** This option sends mouse scrolling through to the Viewer window.
- **Local Window.** This option applies mouse scrolling to the Viewer window. (When the Viewer window does not have scroll bars, mouse scrolling is sent through to the SUT.)

## Mouse Right Click Key and

## Mouse Middle Click Key

These preference set the modifier keys you can hold to change a left click to a right or middle click in the Viewer window. (If you are using a standard mouse, you can just press your right and middle mouse buttons; these settings are only important if you don't have a right or middle mouse button available.)

## Other Live Mode preferences

- **Auto-scroll window in Live Mode.** When this checkbox is selected, the Viewer window scrolls in the appropriate direction when the cursor is moved outside the Viewer window.
- **Track mouse outside of window in Live Mode.** Select this checkbox to make the Viewer window cursor follow your mouse movement outside the Viewer window.
- **Always scale Viewer window proportionally.** Select this checkbox to maintain a true horizontal-to-vertical ratio when you scale the Viewer window size. Deselect to allow disproportionate horizontal and vertical scaling.

## Capture Mode

- **Display capture rectangle border.** Select this checkbox to display a gray border around the Capture Area in Capture Mode. The border can make it easier to see the Capture Area on dark backgrounds.
- **Detect pulsing images when capturing.** Select this checkbox to allow EggPlant to attempt to recognize pulsing images during image capture. In general, this is a useful feature; however, if you are working over a slow network connection, pulsing detection can make your image captures impractically slow.

## VNC preferences

---

These preferences pertain to VNC connections and communication.

## Reverse Connections

Select the Listen For Reverse Connections checkbox to allow SUTs to initiate VNC connections with the local computer running EggPlant.

## VNC Encodings

This is a list of the common encodings in which data is transferred over VNC connections. The encodings appear in order from most compressed to least compressed, and EggPlant always starts at the top of the list and attempts to use the most compressed encoding available.

Most computers experience the best VNC performance by using the fastest (most compressed) encoding available; however, if you feel that your processor's ability to decompress and draw data is more of a limiting factor than your connection speed, you can disable the more compressed encodings.

## Additional VNC Features

- **Copy Rect** allows faster updates in the Viewer window by redrawing only the portions of the SUT that have changed.
- **Rich Cursor** allows faster updates in the Viewer window by receiving cursor updates as screen locations, rather than re-drawn cursors.
- **Rich Clipboards** enables the clipboard exchange of files and rich text between EggPlant and Vine Server.

## Script preferences

---

Script preferences contain the controls for features that can make it easier for you to read and write scripts.

### When a script is run...

This preference determines how EggPlant deals with unsaved script changes when you run a script.

- **Do nothing.** Select this to run scripts without saving changes.
- **Save that script.** Select this to auto-save script changes before running scripts.
- **Save all scripts.** Select this to auto-save all open scripts before running the current script.
- **Ask.** Select this to be alerted to unsaved script changes when you run a script.

**Note:** When a running script calls another script, the called script is read from the currently saved copy. If you have unsaved changes in the called script, they are not executed.

### When a script is modified...

This preference determines what EggPlant does if another user saves changes to a script while you are using it.

- **Ignore.** Select this to keep the script as it currently appears on your desktop, without loading the newly saved version.
- **Reload the script.** Select this to reload your script when EggPlant detects that the saved version has changed.
- **Ask to Reload.** Select this to have EggPlant ask you whether to keep the current version, or load the saved version of your script.

## Reload breakpoints from last save

Select this checkbox to automatically reset saved breakpoints when you open scripts.

## Allow Text Drag and Drop

Drag-and-drop text is enabled in the Script Editor by default. Deselect this checkbox if you do not want to be able to move text by dragging it.

## Default Script Font

This field displays the default font of your text in the Script Editor. (New text and plain text loaded from other scripts are both displayed in this font.) Click the Set button to select a font on your system.

This font is also used in the initial setting of the `folder` global property at the beginning of each script run. (For more information, see [Global Properties](#).)

## Script Log Font

This is the font used to display script results and output in the Log Area of the Run window. Click the Set button to select a font on your system.

## The Indentation pane

These settings control the automatic indention of text in the Script Editor.

### Indent scripts automatically

Select this checkbox to apply indentation when you insert a comment, paste text, or press Return.

### Indent control structures by:

### Indent continuation lines by:

These preferences determine the size of your indentions. Select *Tab(s)* or *Space(s)*, and type the number of tabs or spaces to indent. The Example Script area displays a preview of the way text appears with your current indentation settings.

*Continuation line* refers to a long statement displayed across multiple lines. (To force a line break in a statement, insert a backslash (/).)

## The Tab Key Pop-Up Menu

Choose an item in this pop-up menu to determine the function of Tab in the Script Editor. The tab key can trigger indentation, insert a tab space, or both.

## The Colorization pane

These settings control the automatic color-coding of text in the Script Editor. The Example Script area displays a preview of the way text appears with your current colorization settings

### Enable syntax coloring

Select this checkbox to turn on automatic script colorization.

### Update colors continuously while typing

Select this checkbox to automatically update colors in the current script line as you type. If this feature is turned off, script colors are updated when you insert a comment, paste, save, or press Return.

## Colorization Settings

These settings allow you to customize the colorization of different script elements.

For each script element, select the first checkbox to auto-colorize it in the Script Editor.

To edit a script element's color, click its color well, and select a new color in the standard Colors panel.

Select the Bold checkbox to display the script element in bold text.

## Run Option preferences

---

*Run Option* preferences are the default values of frequently- used run options.

To change these values for an individual script, use the `SetOption` command or set the global property within the script. (For more information, see [Global Properties](#).)

To restore all run options (from all four tabs) to their default value, click the *Restore Defaults* button at the bottom of the preferences panel.

## The Mouse pane

The Mouse settings determine specific mouse behavior in the Viewer window during script execution.

- **Mouse Click Delay.** This value represents the time between the press and release of a mouse click on the SUT.
- **Double-Click Delay.** This value represents the time between the release of the first click and the press of the second click of a `DoubleClick` command.
- **Mouse Move Speed.** This value represents the maximum distance the mouse cursor can move per segment.
- **Mouse Drag Speed.** This value represents the maximum distance the mouse cursor can drag per segment.
- **Mouse Move Delay.** This value represents the delay between segments of mouse movement.



## Movement Mode

This pop-up menu determines the path a mouse cursor takes when it moves from one place to another. Choose *Direct*, *Horizontal Then Vertical*, or *Vertical Then Horizontal*.

## The Keyboard pane

The Keyboard preferences determine specific keyboard behavior in the Viewer window during script execution. (For more information, see Global Properties.)

- **Key-Down Delay.** This value represents the length of time between a key press and release.
- **NextKeyDelay.** This value represents the delay between typed keystrokes.
- **Send shift key down for capitals.** Select this checkbox to have EggPlant hold the SUT's Shift key while typing capital letters. If this checkbox is deselected, EggPlant sends capital-letter key codes to the SUT, without using the Shift key.

## The Screen pane

The Screen preferences determine search time and image tolerance in the Viewer window.

## Image Search Time

The number in this field represents the maximum time that EggPlant searches for an image match in the Viewer window. The *actual* time may be longer than the time specified here, depending on other factors related to the search.

Click the disclosure triangle to reveal or hide the following preferences.

- **Search Delay.** The value in this field is time between one image search and the next.
- **Search Count.** The value in this field represents the number of times EggPlant searches for a single image match. (When the search count value is 1, EggPlant does not perform a full-screen refresh and reposition the mouse before failing. This can reduce delay when searching for images that are not present in the Remote screen, but it can also cause unnecessary failures due to screen artifacts and timing errors.)

**Note:** The Image Search Time is equal to the Image Search Delay multiplied by one less than the Image Search Count. Changing the Image Search Time affects the other two values, and vice versa. Thus, changing the Image Search Time will affect the other two values and vice versa.

## Precise Image Tolerance

The position of this slider determines the level of precision EggPlant requires to match an image with a tolerant search. The number represents the maximum difference between an original pixel's RGB value and the RGB value of a presumed match in the Viewer window.

## Standard Image Tolerance

The position of this slider determines the level of precision EggPlant requires to match an image with a standard

search. The number represents the maximum difference between an original pixel's RGB value and the RGB value of a presumed match in the Viewer window.

**Note:** These image tolerance values affect your searches throughout every script you run in EggPlant. If you need to change these values, it is usually best to do it on a script-by-script basis.

## The System pane

The settings under the System pane allow you to configure how EggPlant interacts directly with the remote system.

### Remote Work Interval

The value in this field represents the minimum delay between two commands that interact with the SUT. (For more information, see [the RemoteWorkInterval, in Global Properties.](#))

### Force Screen Refresh

Select this checkbox to force the Viewer window to redraw the SUT after every line of script executed. (For more information, see [The ForceScreenRefresh, in Global Properties.](#))

## Text preferences

---

These settings define the text platforms that are available for use in text property lists. Each platform includes a Text Engine, and any number of predefined text styles.

### The Platform Pop-Up Menu

The *Platform* pop-up menu contains the names of all of the operating systems for which you can currently create text property lists.

To display and edit the properties of a platform, choose it in the *Platform* pop-up menu.

### Adding a Platform

To define a new platform, click the *Add* button.

The new platform is created with the same initial settings as the platform that was displayed when you clicked the *Add* button

### Removing a Platform

To remove a platform from Text preferences, choose it in the *Platform* pop-up menu and click the *Remove* button. (You cannot remove the default platform unless you name a different default first.)

Platform removal requires confirmation, and it cannot be undone.

## Set As Default

The default platform is the initial value of the [CurrentTextPlatform global property](#) in your scripts.

To set a platform as the default, choose it in the *Platform* pop-up menu and click the *Set as Default* button. The default platform name is displayed in bold text.

## Text Engine

In the Text Engine pop-up menu, you can choose whether you want to search for your text with OCR (optical character recognition) or a text-image generator specific to your SUT. The choices are listed in the table below:

### Text Engines

Name	Operating System	Additional Details
External TIG	Windows	<i>Host</i> : The host name or IP address of the computer that is hosting your TIG. (To use a TIG on the current SUT, type <i>Current SUT</i> .)  <i>Port</i> : The port on which the TIG host communicates. (The default port is 5899.)
Native TIG	Mac OS X	Currently included in EggPlant on Mac OS X only.
OCR Search	Any operating system	
Pango TIG	Linux	See note, below.
Scripted TIG	Custom	<i>Command</i> : The name of the TIG script.

### Note: Installing the Pango TIG

The Pango TIG is not bundled with EggPlant. To install the Pango TIG, download the bundle from [TestPlant downloads](#), and save the download in the */Library/Frameworks* directory, or in the *~/Library/Frameworks* directory of your user account.

## Text Styles

The Text Styles pop-up menu contains all of the defined text styles for the current platform. Each style includes a defined font, size, text color, and background color; it may also include bold, italic, or underlined text. (For information on using styles, see The Find Text Panel.)

To display and edit a text style, choose it in the Text Styles pop-up menu.

**Note:** Typically, a text style corresponds to a particular type of graphical interface element, such as *menu item* or *title bar*.

## Adding a Text Style

To define a new text style, click the *Add* button.

The new text style is created with the same initial settings as the text style that was displayed when you clicked the *Add* button; you can save time by choosing a text style that has some attributes in common with the new text style.

## Removing a Text Style

To remove a text style from the current platform, choose the text style in the *Text Style* pop-up menu and click the *Remove* button. This action requires confirmation, and it cannot be undone.

## Font and Size Fields

In the *Font* and *Size* fields, type or choose a font and size for the current text style.

**Note:** The fonts in the pop-up menu are native to the computer that is running EggPlant. You can always type the names of fonts native to your SUTs.

## Text and Background Color

To edit a text or background color, click the respective color well and select a new color in the Colors panel.

**Tip:** The color picker in the Colors panel is especially useful for EggPlant; it allows you to copy a color from any place in your display, including the Viewer window. To use the color picker, click the magnifying glass in the Colors panel, then click again wherever you see the color you would like to copy.

## Text Attributes

Select the *Bold*, *Italic*, and *Underline* checkboxes to define a text style with those attributes.

## Sharing Text Platforms with Other Users

All of the text platform information you define in Text preferences is stored in a property list within your user account: `~/Library/Eggplant/TextPlatform.plist`.

To share platform definitions with another user account, copy this file into that account's *Library/Eggplant* folder. (This folder is not created until the user changes a text preference.)

## Mail preferences

---

These settings determine the default mail server EggPlant uses for the [SendMail](#) command.

## SMTP Server

In the *SMTP Server* field, enter the host name or IP address of the server you use for sending e-mail.

## Authentication

In the *Authentication* pop-up menu, choose *None*, *Plain*, *Login*, or *CRAM-MD5* as appropriate for your mail server.

## User Name and Password

In these fields, enter the user name and password of the account you use for authentication. If *None* is chosen in the *Authentication* pop-up menu, these fields are not available.

# EggPlant Commands and Functions

The commands and functions defined in this section are an EggPlant extension of the SenseTalk scripting language.

To incorporate the general Sensetalk commands and functions into your scripting, see the *SenseTalk Reference Manual*.

For reference tables of EggPlant commands and functions, see [Appendix A](#) and [Appendix B](#) respectively.

## Command and Function basics

---

Commands and functions are used very similarly. The significant differences between them are:

- Functions always have return values; commands *may* have return values.
- Functions are typically used as parameters to commands.
- Commands must be written alone, not part of a statement; functions must be part of a statement.

## Syntax Guidelines

- 1 Separate command and function parameters with commas:

**Example:** `Click "SomeImage", "SomeOtherImage"`

- 2 Enclose function parameters in parentheses:

**Example:** `ImageLocation ("ThisImage", "ThatImage")`

## Data types used in EggPlant Commands and Functions

---

This section defines the following data types used with EggPlant commands and functions: numbers, coordinates, strings, and property lists.

### Numbers

Single numbers passed as parameters are not enclosed in quotation marks or parentheses.

**Example:** `ScrollWheelUp 10`

### Time

Time parameters are measured in seconds, unless you indicate otherwise. SenseTalk understands the terms *minutes*, *seconds*, and *milliseconds*.

**Examples:** `Wait 75`  
`Wait 1 minute 15 seconds`

## Coordinates

When numbers are given as coordinates, they are enclosed in parentheses.

**Example:** `Click (40,65)`

## Adjusted Coordinates

EggPlant also allows vector arithmetic to generate a location relative to a known position.

**Examples:** `Click ((30,35) + (100,5))`  
`Click (ImageLocation("SomeButton") + (100,5))`

## Strings

Strings, such as file names and text literals, are enclosed in quotation marks.

**Example:** `Click "CancelButton"`  
`TypeText "Here is some text."`

**Note:** Double angle-brackets << >> can enclose text that contains quotation marks and large blocks of text with return characters.

## Property Lists

Property lists define or identify objects by listing certain properties. Each property comprises a key, which identifies what the property represents, and a value, which defines the property for the object.

**Example:** `(ImageName: "MenuButton", HotSpot: (10, 23))`

The example above describes an image that is the same as the saved image called “MenuButton”, but with a different Hot Spot location: (10,23). The key *ImageName* represents the saved image that this property list is based on. The value of *ImageName* is “MenuButton”, the actual name of the image.

Property lists are always enclosed in parentheses, with a colon between each key and its value.

[Appendix D](#) contains tables of the types of property lists used in EggPlant.

For more information, see “Property Lists” in the *SenseTalk Reference Manual*.

## Image references

---

There are several ways to refer to images as command and function parameters: image name, image collection, image property list, and text property list.

### Image Name

Image names are text strings, and must be enclosed in quotation marks. File extensions are optional.

**Example:** `Click "ImageOne"`  
`Click "ImageOne.jpg"`

## Image Collection

An Image Collection is a folder of related images that are all acceptable versions of a basic image. For example, an Image Collection might contain a "normal" image of a button, a selected version, and a disabled version.

Any image folder in your suite can be used as an Image Collection; just keep in mind that EggPlant looks for every image in the Image Collection, so a very large folder could slow down your script execution and cause EggPlant to find matches you are not interested in.

Image Collections are text strings, just like image names.

**Example:** `Click "SaveButton"`

**Note:** The functions `ImageSize()` and `ImageHotSpot()` cannot have Image Collections as parameters.

## Collection Filters

You can use the global property `the CollectionFilter` to narrow down the images EggPlant looks for within a script, by name, image description, or both.

**Example:** `Set the CollectionFilter to (Name: "German")`  
`Set the CollectionFilter to (Name: "German", Description: "New version")`

In the first example, EggPlant would limit searches to images with the string "German" in the image name. In the second example, it would limit searches to images with "German" in the name, and "New version" in the image description (as recorded in the Images pane of the Suite Editor).

**Note:** For filtering purposes, an image's subpath within an Image Collection is considered part of its name.

## Image Property List

An image property list refers to a pre-existing image or Image Collection with the ability to override its default properties. Every image property list must contain the `ImageName` property, and may contain any additional properties from the following list.

- **ImageName:** *file name*. The name of the image file. (Required.)
- **HotSpot:** *coordinate pair*. The Hot Spot coordinates within the image.
- **SearchType:** *Precise, Tolerant, or Text*. A general measure of how closely colors on the SUT must match the colors of your images.
- **Tolerance:** *integer*. A number that represents the acceptable color difference between pixels in an image and a match in the Viewer window. (You can set a single value that is used for red, green, and blue, or three separate values separated by commas.)
- **Discrepancy:** *integer*. The percentage of pixels (with a percent sign) or number of pixels (with no percent sign) by which the image and a match in the Viewer window can differ.
- **Pulsing:** *boolean*. Whether or not the search type allows for a pulsing image.
- **CollectionFilter:** *text string*. For Image Collections only, `CollectionFilter` limits the images that are searched for



within an Image Collection. (For more information, see Collection Filters, above.)

- **ClipRectangle:** *rectangle*. With (0,0) being the top-left corner of the image, the *cliprectangle* is a pair of coordinates that define a rectangle within the captured image. Only the pixels inside this rectangle are considered for image matches.
- **SearchRectangle:** *rectangle*. With (0,0) being the top-left corner of the screen, the *SearchRectangle* is a pair of coordinates that define a rectangle in the Viewer window. EggPlant only looks for this image within the defined rectangle.

**Example:** `Click (ImageName: "CloseButton", HotSpot: (10, 23), SearchType: "Tolerant")`

## Text Property List

A text property list (formerly text-image property list) is a description of text on the SUT. Every text property list must contain the `Text` property, which defines the actual text string you are looking for, and any number of the additional properties described below. Any property that is not included in the property list defaults to your Text preferences settings.

## Common Text Properties

The following text properties can be used with any text property list:

- **Text:** *text string*. The text string that you want to find on the SUT. (Required.)
- **TextPlatform:** *text-platform name*. The name of the text platform that generates the Text Image. (For more information, see [the TextPlatforms Global Property](#).)
- **TextStyle:** *text-style name*. A group of predefined text properties. (For more information, see [The Find Text Panel](#).)

## Generic (OCR) Text Properties

The following text properties can be used with the generic text platform (or any text platform that uses the OCR text engine):

- **CaseSensitive:** *boolean*. Whether or not EggPlant considers case in text searches.
- **TextLanguage:** *language name*. The natural language of the text you are searching for.
- **Contrast:** *boolean*. Whether or not the SUT display is internally converted to a high contrast, two-color image. If contrast is on, the contrast color is considered the primary color of the SUT display, and all other colors are treated as the secondary color. (Text can be found in either color.)
- **ContrastTolerance:** *boolean*. When contrast is on, `contrastTolerance` is the maximum per-channel color difference that is allowed for a pixel to be seen as the contrast color. The default contrast tolerance is 45.
- **ContrastColor:** *color name or value*. When contrast is on, `contrastColor` is the color that is considered the primary color of the SUT display.

## Platform-Specific (TIG) Text Properties

The following text properties can be used with text platforms whose text engine is a text-image generator (TIG):

- **TextFont.** The name of the font used in the Text Image.
- **TextSize.** The size of the text in points.
- **TextColor.** The color of the text.
- **TextBackgroundColor.** The background color of the Text Image.
- **Bold.** Whether the font is displayed as bold; true or false.
- **Italic.** Whether the font is displayed in italics; true or false.
- **Underline.** Whether the text is underlined; true or false.
- **Anti-aliasing (Pango TIG only).** Whether or not text is anti-aliased; on or off.
- **UseMarkup.** Whether or not supported markup tags are recognized as text attributes (on), or treated as string literals (off).

**Example:** `Click (Text: "Cancel", TextStyle: Win2K, TextBackgroundColor: white)`

For more information, see [The Find Text Panel](#) and [Text preferences](#).

## Mouse Commands and Functions

---

This section describes the EggPlant commands and functions that control the SUT's mouse in the Viewer window.

### Click Command

**Example:** `Click "ImageOne", "ImageTwo", "ImageThree"`  
`Click (2,4)`

**Parameters:** One or more image references; or a single coordinate location.

**Behavior:** Clicks the SUT mouse in the Hot Spot of the first location found. (EggPlant searches for images in the order in which they are listed.)

### DoubleClick Command

**Example:** `DoubleClick "SomeImage", (Text:"SomeWords", Italic: No)`  
`DoubleClick (25,358)`

**Parameters:** One or more image references; or a single coordinate location.

**Behavior:** Double-clicks the SUT mouse in the Hot Spot of the first location found. (EggPlant searches for images in the order in which they are listed.)

## RightClick Command

**Example:** `RightClick (imageName: "SomeImage", searchType: Text)`  
`RightClick (12,355)`

**Parameters:** One or more image references; or a single coordinate location.

**Behavior:** Right-clicks the SUT mouse in the Hot Spot of the first location found. (EggPlant searches for images in the order in which they are listed.)

## MouseButtonDown, MouseButtonUp Commands

**Example:** `MouseButtonDown 2`  
`MouseButtonUp 4`

**Parameter:** A single mouse button number, 1-8.

**Behavior:** Presses (MouseButtonDown) or releases (MouseButtonUp) the mouse button indicated by the parameter. These commands are the only way to send mouse button events other than left and right clicks, such as clicks on the middle button or pressing and holding the mouse button for a period of time.

**Note:** For standard 3-button mice, button 1 is the left button, button 2 is the middle button, and button 3 is the right button. For some VNC servers, buttons 4 and 5 control scroll-wheel behavior.

## ScrollWheelDown, ScrollWheelUp Commands

**Example:** `ScrollWheelDown 5`  
`ScrollWheelUp 10`

**Parameter:** A single integer representing mouse wheel increments (audible clicks on some mice.).

**Behavior:** Scrolls the mouse wheel up or down. The actual amount of scrolling per increment varies by mouse driver, platform, and application.

**Tip:** An easy way to experiment with the mouse wheel of a SUT is to run a ScrollWheelUp or ScrollWheelDown command with a parameter of 1.

## MoveTo Command

**Example:** `MoveTo "ImageOne", "ImageTwo"`  
`MoveTo (190,87)`

**Parameters:** One or more image references; or a single coordinate location.

**Behavior:** Moves the SUT's mouse cursor to the Hot Spot of the first image found, or in the location indicated by a coordinate pair.

## MoveToEach Command

**Example:** `MoveToEach "ImageOne", (780, 91), "ImageThree"`

**Parameters:** One or more image references or coordinate locations.

**Behavior:** Moves the SUT's mouse to the locations indicated by the parameters, in the order they are listed.

## Drag Command

**Example:** `Drag "ImageOne", "ImageTwo"`  
`Drag (231, 100)`

**Parameters:** One or more image references; or a single coordinate location.

**Behavior:** Presses and holds the SUT's mouse button in the Hot Spot of the first image found, or in the location indicated by a coordinate pair.

## Drop Command

**Example:** `Drop "ImageOne",`  
`Drop (231, 100)`

**Parameters:** One image reference; or a single coordinate location.

**Behavior:** Moves to the given location and releases the mouse button.

## DragAndDrop Command

**Example:** `DragAndDrop "ImageOne", "ImageTwo", (980, 322)`

**Parameters:** Two or more locations, identified by image references or coordinate pairs.

**Behavior:** Presses and holds the SUT's mouse button in the first location, moves the mouse to subsequent locations, and releases the mouse button in the final location.

## MouseLocation() Function

**Example:** `put MouseLocation() into StartingPoint`

**Returns:** Coordinates of the current mouse location.

**Behavior:** Returns the coordinates of the current mouse location.

## Text Commands and Functions

---

This section describes the EggPlant commands and functions that act upon the SUT's keyboard and clipboard.

## CaptureTextImage Command

**Example:** `CaptureTextImage (text: "some words", rectangle: (0,0)(50,20))`

**Parameters:** One property list that must include a text property and a rectangle property, and may include any other text properties.

**Behavior:** The CaptureTextImage command is only used in scripted text-image generators. Creates a text image of generated text. The text image includes all of the properties you specify in the property list, and fills in remaining properties with default values from Text preferences. (For more information, see [TextStyle Properties under The TextPlatforms](#).)

## KeyDown Command

**Example:** `KeyDown ControlKey, CommandKey, "d"`

**Parameters:** Keyboard characters (in quotation marks); TypeText keywords (with no quotation marks).

**Behavior:** Presses and holds the given keys until they are released by a KeyUp command.

## KeyUp Command

**Examples:** `KeyUp ControlKey, CommandKey, "d"`  
`KeyUp AllKeys`

**Parameters:** Keyboard characters (in quotation marks); TypeText keywords (with no quotation marks).

**Behavior:** Releases keys that have been held by the KeyDown command.

**Note:** You can use the keywords *AllKeys*, *AllModifierKeys*, and *AllNonModifierKeys* to release all keys of those respective groups.

## ReadTable() Function

**Example:** `Log ReadTable ((foundImageLocation(), foundImageLocation() + (200,600)))`

**Parameters:** One rectangle in which you want to read text; and an optional property list that includes any number of the properties listed under the ReadText() function.

**Behavior:** Returns the text of a table as a list. The returned list contains one sublist per row of values, with each sublist containing one value per cell detected within that row. The sublists for the rows may or may not contain the same number of values.

**Note:** The ReadTable function performs best when the bounds of the table are included in the rectangle. If no table is found within the given rectangle, an error is thrown.

## ReadText() Function

**Examples:** `Log ReadText (addressCoordinates)`  
`Log ReadText (rectangle:addressCoordinates, multiLine: true)`

**Parameters:** A property list that includes a rectangle and any number of the properties listed below; or a stand-alone rectangle value that may be followed by a property list.

- **DPI:** *integer*. Dots per inch of the SUT screen. The default DPI is 72.
- **Contrast:** *boolean*. When Contrast is on, the OCR engine treats the ReadText rectangle as a flat, two-color image. The primary color is taken from the top left pixel of the rectangle, or the ContrastColor property. Pixels that fall within the ContrastTolerance value of that color are considered to have that color. All other pixels are assigned the secondary color. Text can be read in either color. The default value of Contrast is off.

- **ContrastColor:** *color*. When Contrast is on, ContrastColor is the color that is treated as the primary color of the ReadText rectangle. If you do not set the ContrastColor property, contrast color is taken from the top left pixel of the rectangle.
- **ContrastTolerance:** *boolean*. When Contrast is on, ContrastTolerance is a measure of how much a pixel can differ from the RGB value of the ContrastColor and still be considered the primary color. The default MonoTolerance is 45.
- **MultiLine:** *boolean*. This property only applies when reading text near a point, as opposed to reading text within a rectangle. When MultiLine is on, the readText function returns the line of text associated with your point, and any subsequent lines that appear to belong to the same text block. When MultiLine is off, the ReadText function only returns the line of text associated with your point. The default MultiLine is off.
- **SingleColumnMode:** *boolean*. When SingleColumnMode is on, the OCR engine presumes that there is only one column of text on the screen. When SingleColumnMode is off, the OCR engine may detect multiple columns and order lines of text accordingly in the return value.
- **TextLanguage:** *language*. This property determines which character sets are valid for your ReadText return value. TextLanguage can include multiple languages, separated by commas, such as "English, French, German". The default TextLanguage is English.
- **Trim:** *boolean*. When Trim is on, the OCR engine trims each edge of the ReadText rectangle until a non-background pixel is encountered. The background color is taken from the top left pixel of the rectangle, or the TrimColor property. The default value of Trim is off.
- **TrimBorder:** *integer*. When Trim is on, TrimBorder is the pixel-width of background that is not trimmed from the ReadText rectangle. The TrimBorder can be set to a negative number, to trim non-background edges from the rectangle. The default TrimBorder is 0.
- **TrimColor:** *color*. When Trim is on, TrimColor is the color that is considered the background of the ReadText rectangle. If you do not set the TrimColor property, the background color is taken from the top left pixel of the rectangle.
- **TrimTolerance:** *integer*. When Trim is on, TrimTolerance is a measure of how much a pixel can differ from the RGB value of the TrimColor and still be considered background. The default TrimTolerance is 0.
- **TrimWhitespace:** *boolean*. When TrimWhitespace is on, all whitespace characters are removed from the beginning and end of returned text. When TrimWhitespace is off, ReadText may return text that starts or ends with whitespace characters. The default TrimWhitespace is on.
- **ValidCharacters:** *string*. The set of characters that may be returned by the ReadText function. The ValidCharacters property overrides the TextLanguage property; characters that are not part of ValidCharacters are not returned by the ReadText function. By default, all characters in your chosen TextLanguage are valid.

**Behavior:** Returns the text in a given screen rectangle.

## RemoteClipboard() Function

**Examples:** `Put RemoteClipboard (5)`  
`Put RemoteClipboard ()`

**Parameters:** Optional maximum wait time

**Returns:** Text contents of the SUT clipboard.

**Behavior:** Returns the current text contents of the SUT's clipboard. (The return is valid only if the clipboard has been used during the current VNC connection.)

**Returning content that was recently placed on the clipboard:** If you include a maximum wait time for the `RemoteClipboard()` function, the function returns clipboard content that was added as a result of the *last action* on the SUT. If no new clipboard content is detected within the given time parameter, an error is raised.

**Returning any content that is on the clipboard:** If you do not specify a maximum wait time, the `RemoteClipboard()` function returns the *last known* contents of the SUT's clipboard.

**Note:** To use the clipboard on a Macintosh SUT, be sure to connect to a Vine desktop server on the active user account. (A Vine system server, which runs as a “root” VNC server across all user accounts, does not have access to the clipboard of any single account.)

## SetRemoteClipboard Command

**Example:** `SetRemoteClipboard "new clipboard contents"`

**Parameters:** One text string.

**Behavior:** Puts given text into the SUT's clipboard. From here, you can paste it like any other text that is cut or copied in the SUT.

**Note:** To use the clipboard on a Macintosh SUT, be sure to connect to a Vine desktop server on the active user account. (A Vine system server, which runs as a “root” VNC server across all user accounts, does not have access to the clipboard of any single account.)

## TypeText Command

**Example:** `TypeText "Some Words"`  
`TypeText <<Some Words`  
`with a Return.>>`  
`TypeText AltKey, "x"`

**Parameters:** Text strings enclosed in quotation marks, keywords for non-character keys, and text literals (blocks of text) in double angle brackets (`<<` `>>`).

**Behavior:** Sends keystrokes to the SUT. If modifier keys are sent, they are released when the entire command has been executed. (To send a modifier that is not released automatically, use the “up” and “down” modifier keywords, such as “ShiftUp” and “ShiftDown”. To send any key that is not released automatically, use the `KeyDown` and `KeyUp` commands.)

## TypeText Keywords

To generate modifier keys and other non-character keys in a script, refer to them by keyword. (See examples below.)

## Examples: Generating modifiers and special keys on the SUT

<code>TypeText return</code>	<code>// Sends a Return keystroke to the SUT.</code>
<code>TypeText ControlKey, AltKey, Escape</code>	<code>// "Holds" the SUT's Control and Alt keys, sends an Escape keystroke, then releases the Control and Alt keys.</code>
<code>TypeText ControlKey, "q"</code>	<code>// "Holds" the SUT's Control key, sends a "q" keystroke, and releases the Control key.</code>

For a complete list of these keywords, see [Appendix C: TypeText Keywords](#).

**Note:** When you are using the TypeText panel to create a command, clicking the *Insert* button automatically encloses your text parameters in quotation marks.

## Image-searching Commands and Functions

These commands and functions search for images in the Viewer window, without performing any actions on the SUT.

### Wait Command

**Example:** `Wait 6 // Waits for 6 seconds`  
`Wait 10 minutes //Waits for 10 minutes`

**Parameters:** A period of time. (The default unit is seconds.)

**Behavior:** Halts the next line of script execution for the given period of time.

### WaitFor Command

**Example:** `WaitFor 6.5, "ImageOne", "ImageTwo"`  
`WaitFor 10 milliseconds, "SomeImage"`

**Parameters:** A maximum wait time and one or more image references. (The default unit of maximum wait time is seconds. You can also specify minutes and milliseconds.)

**Behavior:** Halts the next line of script execution until any one of the image parameters is found in the Viewer window. If no image is found in the maximum wait time, an exception is thrown.

### WaitForAll Command

**Example:** `WaitForAll 6.5, "SomeImage", "SomeImageFolder"`  
`WaitForAll 10 milliseconds, "SomeImage", "SomeImageFolder", "SomeImage2"`

**Parameters:** A maximum wait time and one or more image references. (The default unit of maximum wait time is seconds. You can also specify minutes and milliseconds.)

**Behavior:** Halts the next line of script execution until *all* of the image parameters are found in the Viewer window. If all images are not found in the maximum wait time, an exception is thrown.

**Note:** If an enclosed list of images is passed as a WaitForAll parameter, EggPlant considers that parameter "found" when any one of the items on the list is found. For example:

`WaitForAll "crust", "sauce", ("peppers", "olives")`

This command has three parameters: "crust", "sauce", and a *list* of two other items (enclosed in parentheses.) It can be thought to mean:

<code>WaitFor "crust"</code>	<code>// and at the same time...</code>
<code>WaitFor "sauce"</code>	<code>// and at the same time...</code>
<code>WaitFor "peppers", "olives"</code>	<code>// WaitFor either of these items.</code>



## RefreshScreen Command

**Example:** `RefreshScreen`

**Parameters:** None.

**Behavior:** Forces the Viewer window to update and redraw the SUT. This is not usually necessary; however, if you find that a particular operation produces screen artifacts, you can use this command to eliminate the artifacts.

**Note:** The `ForceScreenRefresh` global property causes the Viewer window to refresh after every script command. (For more information, see [the ForceScreenRefresh, in Global Properties.](#))

## ImageFound() Function

**Example:** `Put ImageFound (10, "SomeImage")`  
`Put ImageFound ("SomeImage")`

**Parameters:** One or more image references; can also take a maximum wait time as the first parameter.

**Returns:** *True* or *False*.

**Behavior:** Searches for the given images in the Viewer window; returns *true* if an image is found, and *false* if no image is not found. (A maximum wait time parameter allows time for an image to be found before *false* is returned.)

**Note:** A maximum time of 0 causes EggPlant to perform an immediate search, without refreshing the Viewer window or repositioning the mouse in a further attempt to find the image. This is the fastest way to search for an image that may not be present.

## ImageLocation() Function

**Example:** `Put ImageLocation ("ImageName")`

**Parameters:** One or more image references.

**Returns:** Coordinates.

**Behavior:** Searches for given images in the Viewer window; returns the coordinates of the first image found. (If no image is found, an exception is thrown.)

## EveryImageLocation() Function

**Example:** `put EveryImageLocation ("SomeImage", "SomeOtherImage", "Image3")`

**Parameters:** One or more image references.

**Returns:** The coordinates of every instance of the given image or images

**Behavior:** Searches the Viewer window for all occurrences of the given images; returns a list of the Hot Spot coordinates for every occurrence of the images. If no image is found, `EveryImageLocation` returns an empty list.

**Tip:** When you call the `EveryImageLocation()` function, you can immediately call *the result* to return `ImageInfo()` for every image represented in `EveryImageLocation()`.

**Example:** `put EveryImageLocation("dog", "cat") //Returns a list of coordinates for every instance of these images.`  
`put the result //Returns imageInfo() for every coordinate pair returned by EveryImageLocation.`

## Image Information Functions

---

The functions in this section return various properties associated with images.

## ImageInfo() Function

**Example:** `Put ImageInfo()` into `ListVariable`

**Parameters:** One or more image references;

**Returns:** An image property list, or *list of image property lists*.

**Behavior:** Returns an image property list for a single image, or a list of image property lists for multiple image references. Depending on the data that is available, the list can contain the following properties:

- **ImageName:** The name of the image file.
- **SearchType:** *Precise, Tolerant, or Text*
- **Tolerance:** A number that represents the acceptable difference between the image and a match in the Viewer window. (You can set a single value that is used for red, green, and blue, or three separate values separated by commas.)
- **Discrepancy:** The percentage of pixels (with a percent sign) or number of pixels (with no percent sign) by which the image and a match in the Viewer window can differ.
- **Pulsing:** Whether or not the search type allows for pulsing; returns a *true* or *false* value.
- **ClipRectangle.** With (0,0) being the top-left corner of the image, the *cliprectangle* is a pair of coordinates that define a rectangle within the captured image. Only the pixels inside this rectangle are considered for image matches.
- **SearchRectangle:** With (0,0) being the top-left corner of the screen, the *SearchRectangle* is a pair of coordinates that define a rectangle in the Viewer window. EggPlant only looks for this image within the defined rectangle.
- **ImagePath:** The full pathname of the image file.
- **HotSpot:** The Hot Spot coordinates within the image.
- **ImageSize:** The image's size in pixels, given as (*width, height*)).
- **CaptureLocation:** Coordinates that indicate the position of the top-left corner of the image when it was captured in the Viewer window.
- **Description:** The image description in the Suite Editor Info panel.

## ImageHotSpot() Function

**Example:** `Click ImageHotSpot ("SomeImage") + (20, 32)`

**Parameters:** A *single* image, excluding Image Collections.

**Returns:** Coordinates.

**Behavior:** Returns the coordinates (x,y) of the image's Hot Spot, relative to the top-left corner of the image.

## ImageSize() Function

**Example:** `put ImageSize ("image")` into `ImageSizeVariable`

**Parameters:** A *single* image, excluding Image Collections.

**Returns:** Size in pixels (*width*, *height*).

**Behavior:** This function returns the size in pixels—given as (*width*, *height*)—of the given image. If the image is not found, an exception is raised.

## ImageColorAtLocation() Function

**Examples:** `put ImageColorAtLocation ("myIcon", (23,1)) // Returns the color at location (23,1) of the given image.`  
`put ImageColorAtLocation ("myIcon", imageHotSpot("myIcon")) // Returns the color at the hotspot of the named image.`

**Parameters:** One image name and one coordinate location.

**Returns:** A color value. (For more information, see "Working with Color", in the *SenseTalk Reference Manual*.)

**Behavior:** Returns the color value of a single pixel in the given image. The coordinates of the pixel are relative to the top-left corner of the image.

## Found-Image Information Functions

---

The functions in this section return additional information about the last image that was found in the Viewer window. They are particularly helpful with the *any* variants of commands and other functions, when you do not necessarily know which image was found.

### FoundImageInfo() Function

**Examples:** `Put FoundImageInfo() //Displays imageInfo for the last image found`  
`Put FoundImageInfo()'s imageName into clickedImage //Puts the name of the last image found into the variable "ClickedImage".`

**Parameters:** None; refers to the last image found.

**Returns:** An image property list.

**Behavior:** Returns an image property list for a single image, or a list of image property lists for multiple image references. Depending on the data that is available, the list can contain the following properties:

- **ImageName:** The name of the image file.
- **SearchType:** *Precise*, *Tolerant*, or *Text*
- **Tolerance:** A number that represents the acceptable difference between the image and a match in the Viewer window. (You can set a single value that is used for red, green, and blue, or three separate values separated by commas.)
- **Discrepancy:** The percentage of pixels (with a percent sign) or number of pixels (with no percent sign) by which the image and a match in the Viewer window can differ.
- **Pulsing:** Whether or not the search type allows for pulsing; returns a *true* or *false* value.
- **ClipRectangle.** With (0,0) being the top-left corner of the image, the *cliprectangle* is a pair of coordinates that define a rectangle within the captured image. Only the pixels inside this rectangle are considered for image matches.
- **SearchRectangle:** With (0,0) being the top-left corner of the screen, the *SearchRectangle* is a pair of coordinates that define a rectangle in the Viewer window. EggPlant only looks for this image within the defined rectangle.

- **ImagePath:** The full pathname of the image file.
- **HotSpot:** The Hot Spot coordinates within the image.
- **ImageSize:** The image's size in pixels, given as (*width, height*).
- **CaptureLocation:** Coordinates that indicate the position of the top-left corner of the image when it was captured in the Viewer window.
- **Description:** The image description in the Suite Editor Info panel.

## FoundImageLocation() Function

**Example:** `Click FoundImageLocation()`

**Parameters:** None; refers to the last image found.

**Returns:** Coordinates

**Behavior:** Returns screen coordinates for the Hot Spot of the last image that was found in the Viewer window.

## FoundImageName() Function (Deprecated)

**Examples:** `Put FoundImageName()` into `variable`  
`Click FoundImageName()`

**Parameters:** None; refers to the last image found.

**Returns:** Image name.

**Behavior:** Returns the name of the last image found.

**Note:** FoundImageName is deprecated. Instead, use FoundImageInfo()'s `imageName` or FoundImageInfo()'s `imagePath`

## FoundImageNumber() Function

**Example:** `If FoundImageNumber()` is `2` then...

**Parameters:** None; refers to the last image found.

**Returns:** One number, representing the image's ordinal position in the parameter list of an *any* command or function.

## Script Commands and Functions

---

These commands and functions allow you to modify Run Options and control the execution of scripts.

### PauseScript Command

**Example:** `PauseScript`

**Parameters:** None.

**Behavior:** Pauses script execution before the next command. The script remains paused until you manually continue, step ahead, or abort. (For more information, see [The Run Window](#).)

**Note:** This command has no effect when EggPlant is run from the command line.

## Run Command

**Example:** `Run "OtherScript" , "parameter1" , "parameter2"`

**Parameters:** One script name.

**Behavior:** Calls another script.

**Tip:** If the other script is located in the same suite as the current script (or an associated helper suite), and its name begins with a letter and contains only letters, digits, and underscores, you can call it as a command in the current script. (You do not have to use the Run command; just the name of the script you are calling, followed by any parameters.)

## RunWithNewResults Command

**Example:** `RunWithNewResults "SomeScript" , "parameter" , "parameter2"`

**Parameters:** One script name, and any parameters it requires to run.

**Returns:** A results property list. (See `ScriptResults`, below.)

**Behavior:** Runs another script that generates its own results log, (as opposed to scripts called with the Run command, which are logged as part of the calling script.) The RunWithNewResults command returns a results property list similar to the `ScriptResults()` function. (For more information, see `ScriptResults`, below.)

If the called script fails, the calling script does not automatically fail or raise an exception. (You can script this behavior based on the returned `Status` property.)

**Tip:** You can use this command to write sophisticated master control scripts, with actions based on the `Status` and `ReturnValue` properties. For example, you can call one script if the previous script succeeds, and call a different script if it fails.

## OpenSuite and CloseSuite Commands

**Example:** `OpenSuite "Suites/SuiteName"`  
`CloseSuite "Suites/SuiteName"`

**Parameters:** One suite name, and optional pathname. (If you do not specify an absolute path, EggPlant searches for the suite in all of the currently open suites, the folder containing the current suite, and the default document directory (designated in General preferences.)

**Behavior:** `OpenSuite` makes another suite available for use during the current run; this availability automatically ends when the script run ends, or when `CloseSuite` is called.

**Note:** Changing the `InitialSuites` global property to access a suite is almost always preferable to using the `OpenSuite` command. Suites that you open with the `InitialSuites` are always the first suites checked for image and script resources, and they are checked in the order in which they are passed. Suites that you open with the `OpenSuite` command are checked after the active suite, and the order in which `OpenSuites` are checked is unpredictable.

## OpenSuites() Function

**Example:** `put OpenSuites()` into `SuitesInUse`

**Parameters:** None

**Returns:** A list of all of the suites that are available to the current script.

**Behavior:** Returns a list of all of the suites that are available to the current script. The list includes the script's own suite, and suites that opened by the `OpenSuite` command.

## SetOption and SetOptions Commands

**Example:** `SetOption scriptlogging, yes`  
`SetOptions (searchrectangle: (2,4,100,4), scriptlogging: yes)`  
`SetOptions JoesOptions`

**Parameters:** One or more global properties and the values you are changing them to. (For more information, see: [Global Properties](#).)

**Behavior:** Changes the value of run options for the current script.

## GetOption(), GetOptions() Functions

**Examples:** `put GetOption (scriptLogging)`  
`put GetOptions(runOptions)`  
`put GetOptions()`

**Parameters:** One or more global properties or global property groups.  
 The global property groups are:

- **RunOptions:** the properties described in [RunOption Global Properties](#).
- **TextOptions:** [CurrentTextPlatform](#) and [TextPlatforms](#)
- **OtherOptions:** [CommandLineOutput](#), [ImageDoctor](#), [ScriptAnimation](#), and [ScriptTracing](#).

**Returns:** A global property list.

**Behavior:** Returns the current value of the given global properties.

**Tip:** If you do not request specific global properties, `GetOption` returns a list of all Run Option global properties and their current values. (For more information, see [Global Properties](#).)

## Hide RunWindow, Show RunWindow Commands

**Example:** `Hide RunWindow`  
`Show RunWindow`

**Parameters:** None.

**Behavior:** Hides or shows the Run window.

**Tip:** You can use the `Hide` command to close the Run window when you do not need to watch a script run, and insert a `Show` command to reveal the Run window for the parts of the Run that you want to see.

## Hide RemoteWindow, Show RemoteWindow Commands

**Example:** `Hide RemoteWindow`  
`Show RemoteWindow`

**Parameters:** None.

**Behavior:** Hides and shows the Viewer window. (The Viewer window does not need to be visible while a script is running.)

## RunningFromCommandLine () Function

**Example:** `if RunningFromCommandLine () then  
    Log "Script was run from command line"  
end if`

**Parameters:** None.

**Behavior:** Returns *yes* if the script run was started by a command line call; returns *no* if the script is being run through the GUI interface.

## TraceScreen On/Off Command

**Example:** `TraceScreen On  
TraceScreen Off`

**Parameters:** *On* or *off*.

**Behavior:** Turns TraceScreen mode on and off. When TraceScreen mode is on, a full image of the Viewer window is captured into the execution log immediately before the execution of each command that acts on the SUT. These screen captures are available in the Log Area of the Suite Editor's Results pane.

**Tip:** Because each image can be quite large, use the `TraceScreen On/Off` commands within a script to capture only the images that are useful to you.

## Reporting Commands

---

These commands insert entries into a script's Log file. (Also see [The ScriptLogging in Global Properties.](#))

### Log Command

**Example:** `Log "Message to be recorded in log file"`

**Parameters:** One or more Log messages.

**Behavior:** Creates an entry in the script's Log file. If you set multiple message parameters, each message is inserted as a separate entry. (If the `ScriptLogging` global property value is *Minimal* or *Silent*, this command does nothing.)

**Tip:** Log entries can make it easier to determine where particular log events are occurring in the run.

### LogWarning Command

**Example:** `LogWarning "Message to be recorded in log file"`

**Parameters:** One or more Log messages.

**Behavior:** Behaves just like the `Log` command described above, but displays the log entries in orange text.

**Tip:** The Results pane of the Suite Editor displays the number of warnings associated with each script run. Warnings are counted even when `ScriptLogging` is set to *Minimal* or *Silent*.

### LogError Command

**Example:** `LogError "messageToBeRecordedInLogFile",`

**Parameters:** One or more Log messages.

**Behavior:** Behaves just like the `Log` and `LogWarning` commands described above, but displays the log entries in red text, and logs the script as a failure.

**Note:** If the `ScriptLogging` global property is set to *Silent*, no error messages are inserted into the log file, but the error count for the run is still be incremented, and the ultimate status of the run is *Failure*.

## ScriptResults() Function

**Example:** `put ScriptResults ("someScript") into currentStatus`

**Parameters:** One script name. (If no script is named, the function defaults to the current script.)

**Returns:** A results property list from every run of the given script.

**Behavior:** Returns a results property list from every run of the given script, in chronological order. Each list contains the following properties:

- **LogFile.** The name and absolute path of the Log file.
- **Errors.** A count of errors logged for that run.
- **Warnings.** A count of the warnings logged for that run.
- **Exceptions.** A count of the EggPlant caught and uncaught exceptions raised.
- **Duration.** The length of time the script ran (or has been running) given in seconds.
- **RunDate.** The date and time the run was started.
- **Status.** The status of the run – *Success*, *Failure*, or *Running*
- **ReturnValue.** The returned value of any return statements.

## SendMail Command

**Example:** `sendmail (to: "testmanager@somecompany.com, tester1@somecompany.com", subject: "Test failed", body: "The very important test script generated an error. The log file is attached.", attachment: logfile)`

**Parameters:** One property list, described below.

**Behavior:** Sends e-mail from within a script. The parameters are contained in a property list, described below:

- **smtp\_host:** The host name or IP Address of the mail server.
- **smtp\_type:** The authentication scheme used by the mail server: *None*, *Plain*, *Login*, or *CRAM-MD5*.
- **smtp\_user:** The user account on the mail server; used if an `smtp_type` is specified.
- **smtp\_password:** The password for login to the mail server; used if an `smtp_type` is specified.
- **smtp\_port:** The port used to connect to the mail server. (The default value is 25, the standard mail port.)
- **To:** One or more addresses, separated by commas. (Required.)
- **From:** The user account sending the message. (Required by some mail servers.)
- **Subject:** The subject line for the message.



- **Body (or Message):** The text of the message.
- **Attachment:** A filename or list of filenames to attach to the e-mail.
- **ReplyTo:** The default address to which a reply is sent.
- **CC:** One or more addresses to which a copy of the message is sent, separated by commas.
- **"Content-Type":** The mime type for the body of the e-mail. To send html e-mail, set the "Content-Type" to "text/html". The default mime type is "text/plain". (Please note that you need quotation marks for both the key and the value of this property.)

## EggPlantVersion Function

**Example:** `log EggPlantVersion()`

**Parameters:** None.

**Returns:** The number of your version of EggPlant.

**Behavior:** Returns the number of your version of EggPlant.

## CaptureScreen Command

**Example:** `CaptureScreen (Name: "ImageFileName", Rectangle: ((67,33), imagelocation: "OtherCorner"))`

**Parameters:** An optional property list.

**Behavior:** Captures a snapshot of the entire Viewer window, or a rectangle indicated in the property list. You can customize this command with a list of any number of the following properties:

- **Name:** An image file name and optional path information. (The default file name is Capture\_Screen, saved in the Results directory.)
- **Rectangle (or Rect).** A pair of diagonal locations indicating a rectangle to capture. (The locations can be screen coordinates, and/or image locations.)
- **Increment.** A value of *yes* or *true* appends an automatically incremented number to the image. (The default value is *no*.)
- **ImageInfo.** A property list of additional information. (For more information, see the `ImageInfo()` function, above.)
- **Tip.** Incrementing is helpful if you are capturing several snapshots within a script (as in a loop or frequently called handler.) The images can all have the same image name, with the incremented number distinguishing them; otherwise, each image would overwrite the previous image with the same name.

## ColorAtLocation() Function

**Examples:** `put ColorAtLocation (2,3) into theColor`  
`if ColorAtLocation (foundImageLocation()) is ColorAtLocation (32,7) then`

**Parameters:** One coordinate location.

**Returns:** The color value of the pixel in the given location. (For more information, see "Working with Color", in the *SenseTalk Reference Manual*.)

**Behavior:** Returns the color value of the pixel in the given location. Coordinates are relative to the top-left corner of the Viewer window. (The top-left corner is (0,0), with the x value increasing toward the right, and the y value increasing toward the bottom.)

## StartMovie Command (currently Mac OS X only)

**Example:** `StartMovie "/User/Documents/MovieFile"`

**Parameters:** A file name for the movie you are capturing, and optional file path.

**Behavior:** Starts recording a movie of the Viewer window. If you do not designate a file path, the movie is saved in the suite's Results directory.

To end the recording, insert a `StopMovie` command into the script.

**Tip:** Movies are recorded with no compression; be sure to allow ample disk space when you are recording movies.

## StopMovie Command (currently Mac OS X only)

**Example:** `StopMovie`

**Parameters:** None.

**Behavior:** Stops a movie recording that was started by the script.

## SUT Commands and Functions

---

These commands and functions pertain to VNC connections and the SUT's native display.

### AllConnectionInfo() Function

**Example:** `put AllConnectionInfo() into HostList`

**Parameters:** None.

**Returns:** A list of connection property lists for all of the SUTs in the Connection List.

**Behavior:** Returns all connection property lists.

### Connect Command

**Examples:** `Connect "192.168.1.110"`

`Connect (serverID: "RedHat-test", portNum: "5902", Visible: "Yes")`

**Parameters:** A host name, IP address, or display name from the Connection List.

**Alternative Parameter:** One connection property list.

**Behavior:** Opens a VNC connection with a SUT, or makes a connected SUT the active SUT.

**Note:** If the SUT is not in your Connection List, or you need to override information in the Connection List, you can pass in a connection property list instead of just a host name, IP address, or display name. The connection properties are described below:

- **Server ID:** The host name, IP address. This is the only required property.
- **PortNum:** The port number used by the VNC server on the SUT. (Default value: 5900.)
- **Password:** The password for the VNC server. (Default value: None.)

- **sshHost:** The host name or IP of a computer hosting an SSH connection. (No default.)
- **sshUser:** The user account on the SSH host (No default.)
- **sshPassword:** The password to the user account on the SSH server. (No default.)
- **Visible:** Whether or not the Viewer window automatically opens upon connection; Yes or No.
- **ColorDepth:** The color depth of the SUT in the Viewer window: 8, 16, 32. (Default value: The native color depth of the SUT.)

**Tip:** If you have already connected to this SUT, you can generate this property list with the `ConnectionInfo()` function.

## ConnectionInfo() Function

**Example:** `Connect ConnectionInfo ("SomeSUT")`

**Parameters:** A host name, IP address, or display name of a SUT; a connection property list that contains at least one of the above; or no parameter.

**Returns:** A connection property list for the given connection. (If there is no parameter, `ConnectionInfo` returns a connection property list for the active connection.)

**Behavior:** Returns a connection property list for the given SUT.

The property list returned can include any number of the following properties:

- **Availability:** The last-known availability status of the SUT.
- **Server ID:** The host name, IP address, or display name of the SUT.
- **PortNum:** The port number used by the VNC server on the SUT.
- **Pass\_code:** The password for the VNC server, encrypted.
- **sshHost:** The host name or IP of a computer hosting an SSH connection
- **sshUser:** The user account on the SSH host
- **sshPass\_code:** The password to the user account on the SSH server, encrypted.
- **Visible:** Whether or not the Viewer window automatically opens upon connection; Yes or No.
- **Status:** The status of the connection: *Connected* or *Not Connected*.

## Disconnect Command

**Example:** `Disconnect (serverID:"192.168.1.20", port:"5900")`  
`Disconnect "Nightly Regression"`  
`Disconnect`

**Parameters:** An optional VNC connection, identified by the SUT's name or connection property list.

**Behavior:** Closes the VNC connection with the given SUT. If no SUT is specified, closes the active connection.

## HighlightRectangle command

**Example:** `HighlightRectangle ((FoundImageLocation(), FoundImageLocation() + (50,10))  
HighlightRectangle ((FoundImageLocation(), FoundImageLocation() + (50,10)),  
white)`

**Parameters:** One rectangle value

**Behavior:** Highlights the given rectangle in the Viewer window.

## RemoteScreenRectangle() function

**Example:** `put RemoteScreenRectangle()`

**Parameters:** None.

**Returns:** Coordinates.

**Behavior:** Returns coordinates for a rectangle that shows the full size of the Viewer window.

## RemoteScreenSize() function

**Example:** `put RemoteScreenSize()`

**Parameters:** None.

**Returns:** The width and height of the Viewer window, measured in pixels.

**Behavior:** Returns the size of the Viewer window as a list of two numbers indicating the screen width and height.

## Global Properties

In SenseTalk, global properties are predefined variables that determine behaviors of the run environment.

This section describes the two kinds of global properties that are specially defined for EggPlant: EggPlant global properties, and Run Option global properties.

EggPlant global properties pertain to the EggPlant application in general, and Run Option global properties pertain to interactions with a SUT in the course of a test run.

Many of these global properties appear in the Text and Run Option preferences; using them as global properties allows you to change them on a script-by-script basis, or even multiple times within a single script.

the *SenseTalk Reference Manual* provides more in-depth information about global properties, including a number of global properties that are not EggPlant-specific, but still useful in EggPlant scripting. (For example, the `colorFormat` global property.)

## Using Global Properties with SenseTalk Commands

When you set a global property value with a SenseTalk command, such as `Set` or `Put`, use the word *the* before it the global property to distinguish it from an ordinary variable. (For more information, see the *SenseTalk Reference Manual*.)

**Example:** `set the searchrectangle to (1,2,2,3)`

## Using Global Properties with SetOption and SetOptions

When you call the `setOption` command to set a global property value, do not insert the word *the* before the name of the global property. (`SetOption` is an EggPlant command that takes *only* global properties, so there is no need to distinguish a global property with *the*.)

To set multiple global properties with the `setOptions` command, pass a global property list in parentheses, as you would always pass a list.

**Examples:** `setoption searchrectangle, (1,2,2,3)`  
`setoptions (searchrectangle: (1,2,2,3), scriptlogging: yes)`

**Tip:** To restore all settings to their default value, call `SetOptions` with the parameter *OriginalSettings*.

## GetOption() and GetOptions() Functions

These functions return the value of a global property (`getOption`), or list of global properties (`getOptions`). Enclose single global properties and global property lists in parentheses, as you would for any function.

**Example:** `put getOptions (searchrectangle)`  
`put getOptions (searchrectangle, scriptlogging)`

## EggPlant Global Properties

---

These properties pertain to the EggPlant application in general.

### The CollectionFilter

**Example:** Set **the CollectionFilter** to **(Description: "2007")**  
Set **the CollectionFilter** to **(Name: "English", Description: "release 5.0")**

**Values:** A property list that contains one or both of the properties Name and Description.

**Default:** No filter.

**Description:** Limits image searches to those images whose names or descriptions contain the specified string.

**Tip:** To reset the CollectionFilter, set the value to empty.

**Example:** Set **the CollectionFilter** to **empty**

### The ImageDoctor

**Examples:** Set **the ImageDoctor** to **Manual**  
**SetOption ImageDoctor, Off**

**Value:** *Auto*, *Manual*, or *Off*

**Default:** *Manual*

**Description:** The ImageDoctor global property determines how the Image Doctor works during script execution.

- **Auto.** The Image Doctor automatically attempts to correct image failures and continue the script execution. (The initial image failure is logged as a warning.)
- **Manual.** The Image Doctor panel opens when an image fails; you can choose to proceed with a corrected image, or allow the script to fail.
- **Off.** The Image Doctor is not used during script execution.

### The InitialSuites

**Example:** Set **the InitialSuites** to **( "WindowsXP", "WindowsVista" )**

**Value:** One or more suite names.

**Default:** None.

**Description:** The InitialSuites global property determines the first suite or suites that are searched for script or image resources called by the current script. (Suites named in the InitialSuites take precedence over the calling script's own suite.)

### The FinalSuites

**Examples:** Set **the FinalSuites** to **( "Fallback" )**  
**SetOption ImageDoctor, Off**

**Value:** One or more suite names.

**Default:** None.

**Description:** The `FinalSuites` global property determines the suite or suites that are searched for script or image resources after the script's own suite is searched. (Suites named in the `InitialSuites` are searched first, followed by the script's own suite, and then the `FinalSuites`.)

## The SearchRectangle

**Example:** `setoption SearchRectangle, ((10, 12, 100, 4))`

**Values:** Two image locations or two pairs of coordinates indicating diagonal corners of the searchrectangle. (Regardless of the form you pass in, the return value is displayed as `(x,y,x,y)`.)

**Default:** The full Viewer window

**Description:** Limits the area that EggPlant searches for images in the Viewer window.

**Tip:** To reset the `searchRectangle` to the full Viewer window, set the value to `empty()` or `fullscreen`.

## The TextPlatforms

**Example:** `Put GetOption (TextPlatforms)`

**Value:** A text platform property list.

**Default:** Set in Text preferences.

**Description:** The `TextPlatforms` global property contains the following properties for all of your text platforms:

- **Name:** Name of the text platform.
- **Generator:** The text-image generator (TIG) used by the platform.
- **Styles:** The properties of each of the platform's defined text styles.

## Text Style Properties

For each text style, the following properties are available:

- **TextFont.** The text's font, such as *Helvetica*
- **TextSize.** The text's size, in points.
- **TextColor.** By default, this is an RGB value. For more information, see "Working with Color" in the *SenseTalk Reference Manual*.)
- **TextBackgroundColor.** By default, an RGB value indicating the text's background.
- **Italic.** A *yes* or *no* value.
- **Bold.** A *yes* or *no* value.
- **Underline.** A *yes* or *no* value.
- **Trim.** Whether or not extra space is trimmed away from the text. (The default value is *yes*.)
- **Cache.** Whether a cached image is used, if one is available. (A value of *no* forces EggPlant to generate the image over again every time it is called. The default value is *no*.)

## The CurrentTextPlatform

**Examples:** `setoption currenttextplatform, "WinXP"`  
`set the currenttextplatform.generator to "native"`

**Value:** The name of a text platform defined in Text preferences.

**Default:** Set in Text preferences.

**Description:** The CurrentTextPlatform determines the text platform that is used by a script.

**Note:** This global property *displays* only the name of the current platform by default, but it is still a property list. You can assign values to CurrentTextPlatform keys just as you can with the TextPlatforms global property.

**Tip:** To display the property list of the CurrentTextPlatform, call the SenseTalk function StandardFormat().

## The DefaultUseMarkup

**Examples:** `setoption DefaultUseMarkup, Yes`  
`set the defaultUseMarkup to No`

**Value:** yes or no

**Default:** No

**Description:** The DefaultUseMarkup determines whether or not supported text markups are recognized in the Text property of a text property list. When the DefaultUseMarkup is set to yes, supported markups are used to format generated text images. When the DefaultUseMarkup is set to no, markups are treated as literal text. (Markups are always treated as literal text for text platforms that use the OCR text engine.)

For more information, see "Using Marked-Up Text", in Using EggPlant.

## The DefaultTextStyle

**Examples:** `setoption DefaultTextStyle, "StyleOne"`  
`set the defaultTextStyle.TextFont to "Helvetica"`

**Value:** A text-style property list

**Default:** The default text style of the current platform.

**Description:** The DefaultTextStyle is a shortcut to the default text style for the current text platform. Changes you make here are reflected in the CurrentTextPlatform global property, and vice versa.

## The RemoteClipboard

**Examples:** `set the RemoteClipboard to "Hello"`  
`setOption RemoteClipboard, "Hello"`

**Value:** Contents of the remote clipboard.

**Default:** None.

**Description:** The RemoteClipboard provides access to the clipboard of the SUT. By modifying this global property, you change the contents of the SUT's clipboard.

**Tip:** You can also view and modify the SUT's clipboard with the Remote Clipboard() function and the SetRemoteClipboard command.



## The RepositionPoint

**Example:** `SetOption RepositionPoint, (1280, 1024)`

**Value:** One pair of screen coordinates.

**Default:** A dynamic point near the lower-right corner of the Viewer window.

**Description:** The `RepositionPoint` determines where the mouse is repositioned during image searches.

Positive coordinate values are relative to the top left corner of the screen; negative coordinate values are relative to the bottom right corner of the screen. (See also [the ShouldRepositionMouse](#).)

## The ScriptLogging

**Examples:** `Set the ScriptLogging to Silent`  
`SetOption Scriptlogging, On`

**Values:** *On* (or *true*), *off* (or *false*), *minimal*, or *silent*

**Default:** *On*.

**Description:** The `ScriptLogging` controls which events are recorded in the Log file as a script runs, as follows:

- **On:** Log, LogWarning, and LogError commands, and all interactions with the SUT are recorded.
- **Off:** Only Log, LogWarning, and LogError commands are recorded..
- **Minimal:** Only LogWarning and LogError commands are recorded.
- **Silent:** Nothing is recorded in the Log file. (Warnings and errors are still counted.)

(For more information, see [The Results pane, in The Suite Editor](#).)

## The ScriptAnimation

**Examples:** `SetOption ScriptAnimation, All`  
`Set the ScriptAnimation to Off`

**Value:** *All* (or *true*), *Off* (or *false*), or *Calls*

**Default:** *Off*.

**Description:** The `ScriptAnimation` determines whether or not a script is animated (highlighted line-by-line in the Run window) as it runs, as follows:

- **All or true.** Each line of the script is highlighted as it is executed.
- **Off (or false).** The script is not animated.
- **Calls.** Each script or handler is highlighted, but not each line.

The `ScriptTracing` reverts to its default value upon completion of the script execution. (For more information, see [Animation](#).)

## The ScriptTracing

**Examples:** `SetOption ScriptTracing, Calls`  
`Set the ScriptTracing to Off`

**Value:** *All* (or *true*), *Off* (or *false*), or *Calls*

**Default:** *Off*.

**Description:** The `ScriptTracing` determines whether or not a script is traced (displayed line-by-line in the Run window, one step ahead of actual execution.) as it runs. The values are described below:

- **All or true.** Each line of the script is displayed before it is executed.
- **Off (or false).** The script is not traced.
- **Calls.** Each script or handler is traced, but not each line.

The `ScriptTracing` reverts to its default value upon completion of the script execution.

## The CommandLineOutput

**Examples:** `Set the CommandLineOutput to On`  
`SetOption CommandLineOutput, Off`

**Value:** *On* (or *true*), *Off* (or *false*)

**Default:** *Off*.

**Description:** The `CommandLineOutput` controls the output of messages to the standard output stream when a script is run from the Unix or Windows command line:

- **Off (or False).** No output is sent to standard output (stdout).
- **On (or True).** All of the script output that would appear in the EggPlant graphical interface is sent to the standard output stream.

## Run Option Global Properties

---

These global properties determine interactions with the SUT. The default values are appropriate for most situations.

### The RemoteWorkInterval

**Example:** `SetOption RemoteWorkInterval, 1`

**Value:** Time (in seconds.)

**Default:** 0.7

**Description:** The `RemoteWorkInterval` determines the minimum time EggPlant allows for the SUT to perform a task before sending the next event from a script. This is an important timing parameter for regulating the overall speed of interactions with the SUT.

### The ImageSearchTime

**Example:** `Set the ImageSearchTime to 5`

**Value:** Time (in seconds.)

**Default:** 1.8

**Description:** The `ImageSearchTime` determines the length of time EggPlant searches for an image before reporting a failure.

## The ImageSearchCount

**Example:** `SetOption ImageSearchCount, 3`

**Value:** A positive integer.

**Default:** 7

**Description:** The `ImageSearchCount` determines the number of times EggPlant scans the Viewer window searching for an image.

**Note:** Setting the `ImageSearchCount` to 1 causes image searches to look at only the current Viewer window, bypassing any `ImageSearchDelay`, screen refreshes, and mouse repositioning. This is the fastest way to perform a search, but it can be prone to failure.

## The ImageSearchDelay

**Example:** `Set the ImageSearchDelay to .6`

**Value:** Time (in seconds.)

**Default:** 0.3

**Description:** The `ImageSearchCount` determines the wait time between searches for an image.

## The KeyDownDelay

**Example:** `SetOption KeyDownDelay, .05`

**Value:** Time (in seconds.)

**Default:** 0.001

**Description:** The `KeyDownDelay` determines the wait time between `keyDown` and `KeyUp` events sent to the SUT.

## The NextKeyDelay

**Example:** `SetOption NextKeyDelay, .08`

**Value:** Time (in seconds.)

**Default:** 0.02

**Description:** The `NextKeyDelay` determines the wait time between keystrokes sent to the SUT.

## The ReadTextSettings

**Example:** `Set the readTextSettings to (dpi:96, multiLine: off)`

**Value:** a property list

**Default:** (dpi: 72)

**Description:** The `ReadTextSettings` is a property list of the default settings used for the `ReadText` and `ReadTable` functions.

## The SendShiftForCaps

**Example:** `Set the SendShiftForCaps to yes`

**Value:** Yes, Always, and Never.

**Default:** Yes.

**Description:** The `SendShiftForCaps` determines whether or not `ShiftKeyDown` is used for capital letters sent to the SUT.

- **Yes.** Sends the shift key to type capital letters in commands that do *not* include a modifier-key parameter. For example, `TypeText AltKey & "S"` would not send a shift to generate a capital S, because doing so would change the result from `Alt+s` to `Shift+Alt+s`. However, `TypeText "Seattle"` would send a shift to type a capital S.
- **Always.** Sends the shift key to the SUT whenever a capital letter appears in a `TypeText` parameter.
- **Never.** Never sends shift to type a capital letter. (You can still send the shift key by specifically passing the `ShiftKey` parameter.)

## The MouseClickDelay

**Example:** `SetOption MouseClickDelay, .5`

**Value:** Time (in seconds.)

**Default:** 0.02

**Description:** The `MouseClickDelay` determines the wait time between `mouseDown` and `mouseUp` events sent to the SUT.

## The MouseDoubleClickDelay

**Example:** `Set the MouseClickDelay to .004`

**Value:** Time (in seconds.)

**Default:** 0.01

**Description:** The `MouseDoubleClickDelay` determines the wait time between the end of the first click (`mouseUp` event) and the beginning of the second click (`mouseDown` event) of double-clicks sent to the SUT.

## The MouseMoveSpeed

**Example:** `SetOption MouseMoveSpeed, 10`

**Value:** A number, 0 or greater.

**Default:** 0

**Description:** The `MouseMoveSpeed` determines how fast the SUT's mouse is moved in the Viewer window. The value represents the number of pixels EggPlant moves the SUT's mouse in each step. (Between steps, there is a pause the length of the `MouseMoveDelay` value.)

With the default value of 0, the mouse jumps from its current position to the next point in one step.

**Note:** Some systems (particularly certain Linux systems) have trouble tracking the mouse under the default value. For these systems, start with a value of 10, then reduce the value as necessary until you find one that works. (Conversely, if 10 is slow enough, you can try increasing the value to see how fast the mouse can move before the SUT can no longer track it.)

## The MouseDragSpeed

**Example:** Set `the MouseDragSpeed` to `12`

**Value:** A number, 0 or greater.

**Default:** 10

**Description:** The `MouseDragSpeed` determines how fast the SUT's mouse is moved during a drag command. The value represents the number of pixels EggPlant drags the SUT's mouse in each step. (Between steps, there is a pause the length of the `MouseMoveDelay` value.)

## The MouseMoveDelay

**Example:** Set `the MouseMoveDelay` to `.06`

**Value:** Time (in seconds.)

**Default:** 0.01

**Description:** The `MouseMoveDelay` determines the wait time between each step of a SUT's mouse move.

## The MouseMoveMode

**Example:** `SetOption MouseMoveMode, 1`

**Value:** 0, 1, or 2

**Default:** 0

**Description:** The `MouseMoveMode` determines the path a mouse takes to move from one location to the next, as follows.

- 0. The mouse moves in a straight line.
- 1. The mouse moves horizontally, then vertically.
- 2. The mouse moves vertically, then horizontally.

**Note:** If the `MouseMoveSpeed` value is 0, the mouse jumps to the next location instantly, and the `MouseMoveMode` value is irrelevant.

## The ShouldRepositionMouse

**Example:** Set `the ShouldRepositionMouse` to `True`

**Values:** *True* or *false*.

**Default:** *True*.

**Description:** The `ShouldRepositionMouse` determines whether or not the SUT mouse moves to the lower-right corner of the Viewer window (or the coordinates of `the RepositionPoint`) during image searches. (Repositioning the mouse prevents it from obscuring images elsewhere.)

## The StandardImageTolerance/ PreciseImageTolerance

**Example:** `Set the StandardImageTolerance to 50`  
`SetOption PreciseImageTolerance, 4`

**Value:** An integer, 0-255.

**StandardImageTolerance Default:** 45

**PreciseImageTolerance Default:** 1

**Description:** These global properties determines how close an image's color must be to a color in the Viewer window for EggPlant to consider it a match by tolerant or precise search type standards. (The value represents the difference between an RGB value in the image, and the value of its corresponding location in the Viewer window.)

**Note:** The default values are appropriate for most situations. Practical values can range from 0 to 100, but 80 is often the point that starts to generate a lot of false matches.

## The ForceScreenRefresh

**Example:** `SetOption ForceScreenRefresh, true`

**Default:** *False*.

**Description:** The ForceScreenRefresh determines whether or not EggPlant refreshes the Viewer window after each command.

**Note:** The ForceScreenRefresh is rarely necessary, and dramatically slows down script execution; however, it might be helpful with a SUT that is generating a lot of screen artifacts or running software that does not display well through the VNC server.

## Running from the Command Line

There are several reasons why you might want to run scripts or suites from the command line:

- You can have one or more instances of EggPlant running tests from the command line while you continue to develop and run scripts in GUI mode.
- You can add a command line call to EggPlant at the end of a product build script to perform a smoke test of the new build.
- You can use the Unix `crontab` or `at` commands (built into Mac OS X) to schedule EggPlant scripts to run at specific times.

**Note:** Every instance of EggPlant requires a license. To use EggPlant from the GUI and the command line at once, you need at least two licenses.

## Running Scripts from the Command Line

The *runscript* command launches EggPlant and runs scripts without showing the EggPlant GUI. (To control the full-GUI EggPlant through the command line, see [GUI Options in the Command Line](#), below.)

- 1 Type the full path of the EggPlant application, followed by */runscript*, and a space.
- 2 Type the full path of a script or suite to run.
- 3 For each additional script or suite, insert a space, then type the full path to the script or suite.

**Example:** `/Applications/Eggplant.app/runscript /Users/Alex/appTests.suite  
/Applications/Eggplant.app/runscript /Users/Alex/appTests.suite/scripts/test1.  
script`

EggPlant saves the results of each script run in the Results folder for that script's suite. Then, after all scripts and suites have finished running, EggPlant reports the number of successful scripts and suites as the return value. (A suite is successful if every script in it is successful.)

**Note:** To have EggPlant report script failures instead of successes, set the [-ReportFailures](#) option to yes.

### Passing Parameters

You can pass parameters to a single script, or the last of multiple scripts to be run. At the end of the runscript line, type *-params*, and then the parameters, separated by spaces.

**Example:** `/Applications/Eggplant.app/runscript /Users/Alex/appTests.suite/Scripts/  
test1.script -params "SUT1" "phonebook.txt"`

Each parameter is passed as a text string.

### Runscript Options

The options described below can be added to a runscript line. To add multiple options, insert a space before each one.

#### -CommandLineOutput

**Use:** Controls whether script logs (the data shown in the log area of the Run window) are output. To output script logs, set the *CommandLineOutput* global property to yes. To turn off this output, set the *CommandLineOutput* global property to no.

**Example:** `/Applications/Eggplant.app/runscript /Users/Alex/appTests.suite/scripts/  
test3.script -CommandLineOutput NO`

You can also set the [CommandLineOutput\(\)](#) global property within a script. Global properties that are changed in a script always override initial settings.

**Note:** Output set to standard error (stderr) is always displayed. For finer control over the output while running from the command line, you can use this syntax in your script:

`write "THIS IS SOME OUTPUT" to error`



## Connection Information (-host, -port, -password, -colorDepth)

**Use:** The -host argument is the server ID of a specific SUT to use for your scripts. When you pass in a -host argument, you can also specify a port, password, and color Depth.

**Example:** `/Applications/Eggplant.app/runscript /Users/Alex/Docs.suite/scripts/myTests/test2.script -host SUT1 -port 5901 -password broncos colorDepth 16`

**Note:** A Connect command from within a running script or connection information in a schedule overrides this connection. (This connection remains open, but the new connection becomes the active connection.)

## -GlobalResultsFolder

**Use:** Sets the name of the folder in which your script results are stored. The folder path can be absolute, or relative to the script's suite.

**Example:** `/Applications/Eggplant.app/runscript /Users/Alex/Docs.suite/scripts/test2.script -GlobalResultsFolder ~/MyResults`

## -repeat

**Use:** Repeats the scripts you are running, x times.

**Example:** `/Applications/Eggplant.app/runscript /Users/Alex/myTests.suite -repeat 4`

## -ReportFailures

**Use:** Reports the number of *failed* scripts and suites back to the command line after script execution. If you do not set -ReportFailures to yes, EggPlant reports the number of *successful* scripts and suites to the command line.

**Example:** `/Applications/Eggplant.app/runscript /Users/Alex/Docs.suite -ReportFailures yes`

## Launching the EggPlant GUI from the Command Line

---

To launch the full GUI EggPlant application from the command line, type the path to the EggPlant executable.

**Example:** `/Applications/Eggplant.app/Contents/MacOS/Eggplant`

## Command Line Options

### -AlertOnError

**Use:** Overrides the EggPlant General Preference *Show alert when a connection fails*. (Takes a yes/no value.)

**Example:** `/Applications/Eggplant.app/Contents/MacOS/Eggplant -AlertOnError yes`

### -AutoShowRunWindow

**Use:** Overrides the EggPlant General Preference *Bring the Run window forward when a script is run*. Requires a yes/no value.

**Example:** `/Applications/Eggplant.app/Contents/MacOS/Eggplant -AutoShowRunWindow yes`

## -QuitAfterRun

**Use:** Runs the given script or suite, then quits EggPlant. Requires a yes/no value. (A no value has the same result as omitting this option.) To cancel the quit, choose EggPlant menu > Quit, then click the Cancel Quit button.

**Example:** `/Applications/Eggplant.app/Contents/MacOS/Eggplant -QuitAfterRun yes`

## -RestoreConnection

**Use:** Overrides the EggPlant General Preference *Reopen connections from last session*. (Takes a yes/no value.)

**Example:** `/Applications/Eggplant.app/Contents/MacOS/Eggplant -RestoreConnection yes`

## -ReopenDocuments

**Use:** Overrides the EggPlant General Preference *Reopen suites and scripts from last session*. (Takes a yes/no value.)

**Example:** `/Applications/Eggplant.app/Contents/MacOS/Eggplant -ReopenDocuments no`

## -RunOnLaunch

**Use:** Runs the given scripts or suite when the EggPlant application is launched.

**Example:** `/Applications/Eggplant.app/Contents/MacOS/Eggplant -RunOnLaunch /Users/Alex/  
Alex_Tests.suite`

## Appendix A: EggPlant Commands

### Eggplant Commands

Command	Behavior
CaptureScreen	Takes a snapshot of the Viewer window
CaptureTextImage	Creates a text image. This command is only used in scripted text-image generators.
Click	Clicks the SUT mouse in a coordinate location, or in the first image parameter found
CloseSuite	Ends the temporary suite availability of OpenSuite
Connect	Opens a VNC connection with a SUT; makes it the active connection
Disconnect	Closes a VNC connection with a SUT
DoubleClick	Double-clicks the SUT mouse in a coordinate location, or in the first image parameter found.
Drag	Clicks and holds the SUT mouse in the Hot Spot of the first location found
DragAndDrop	Clicks and holds the SUT mouse in the first location, moves to subsequent locations, releasing the mouse button on the last location
Drop	Moves the mouse to the given location and releases the mouse button
Hide RemoteWindow	Hides the Viewer window
HighlightRectangle	Draws a border around the given rectangle in the Viewer window
KeyDown	Presses and holds keys on the SUT keyboard.
KeyUp	Releases keys that are held down on the SUT keyboard.
Log	Inserts a comment into a script's Log file
LogError	Inserts a comment into a script's Log file in red text
LogWarning	Inserts a comment into a script's Log file in orange text
MouseButtonDown	Presses the SUT mouse button
MouseButtonUp	Releases the SUT mouse button
MoveTo	Moves the SUT mouse to a coordinate location, or to the first image parameter found
MoveToEach	Moves the SUT mouse to each of the given coordinate locations and image parameters
OpenSuite	Makes a suite available to the running script
PauseScript	Pauses script execution; enters debug mode
RefreshScreen	Redraws the image of the SUT in the Viewer window
RightClick	Right-clicks the SUT mouse in a coordinate location, or in the first image parameter found
Run	Runs a given script
RunWithNewResults	Runs a given script, generating a separate Log file for it
ScrollWheelDown	Scrolls the SUT mouse wheel down a given number of increments
ScrollWheelUp	Scrolls the SUT mouse wheel up a given number of increments
SendMail	Sends an e-mail from within a script
SetOption	Modifies a global property value
SetOptions	Modifies multiple global property values
SetRemoteClipboard	Places text in the SUT clipboard
ShowRemoteWindow	Shows the Viewer window

Command	Behavior
StartMovie	(Currently Mac OS X only.) Starts recording a movie in the Viewer window
StopMovie	(Currently Mac OS X only.) Ends a movie that is recorded from within a script
TraceScreenOn / TraceScreenOff	Turns screen captures before every command on and off.
TypeText	Sends keystrokes to the SUT keyboard
Wait	Delays the next line of execution for the given length of time
WaitFor	Delays execution of the next line until any one of the given images is found
WaitForAll	Delays execution of the next line until all given images are found.

## Appendix B: EggPlant Functions

### EggPlant Functions

Function	Returns
<a href="#">ColorAtLocation()</a>	The color of the pixel at a given location (Default: RGB value)
<a href="#">ConnectionInfo()</a>	A connection property list
<a href="#">EggPlantVersion()</a>	The version number of your copy of EggPlant
<a href="#">EveryImageLocation()</a>	Coordinates of every occurrence of given images
<a href="#">FoundImageInfo()</a>	An image property list for the last image found
<a href="#">FoundImageLocation()</a>	The location of the last image found
<a href="#">FoundImageName()</a>	Name of the last image found. (This function is deprecated. See <a href="#">foundImageInfo()</a> .)
<a href="#">FoundImageNumber()</a>	The position of the last image found, within a series of given images
<a href="#">GetOption()</a>	The value of a given global property
<a href="#">GetOptions()</a>	The values of all Run Option global properties or given global properties
<a href="#">ImageColorAtLocation()</a>	The color of the pixel at a given location within an image (Default: RGB value)
<a href="#">ImageFound()</a>	Whether a given image is found; <i>True</i> or <i>false</i>
<a href="#">ImageHotSpot()</a>	The location of the Hot Spot in the given image
<a href="#">ImageInfo()</a>	An image property list for the given image
<a href="#">ImageLocation()</a>	The location of the given image
<a href="#">ImageSize()</a>	An image width and height in pixels
<a href="#">AllConnectionInfo()</a>	A list of connection property lists for every SUT in the Connection List
<a href="#">MouseLocation()</a>	Coordinates of the current mouse cursor location
<a href="#">OpenSuites()</a>	A list of all suites available to the current script
<a href="#">ReadTable()</a>	The table data found within a given rectangle on the SUT
<a href="#">ReadText()</a>	The text content found within a given rectangle on the SUT
<a href="#">RemoteClipboard()</a>	Contents of the SUT clipboard
<a href="#">RemoteScreenRectangle()</a>	The coordinates of a rectangle the size of the entire Viewer window
<a href="#">RemoteScreenSize()</a>	The width and height of the Viewer window in pixels
<a href="#">RunningFromCommandLine()</a>	Whether or not the current script run was called in a command line
<a href="#">ScriptResults()</a>	A results property list for every run of the given (or current) script

## Appendix C: TypeText Keywords

### TypeText Keywords

To press this key...	Type this keyword (without quotation marks)
0, 1...9 (on Keypad)	keypad0, keypad1...keypad9
Alt (Windows; For Mac, use Option)	AltKey
Application key (Windows)	applicationKey
Arrow down	downArrow
Arrow left	leftArrow
Arrow right	rightArrow
Arrow up	upArrow
Backslash (\)	backslash
Backspace/Delete	backspace
Begin	beginKey
Command	CommandKey
Command down	CommandDown
Command up	CommandUp
Control	ControlKey
Control down	ControlDown
Control up	ControlUp
Decimal (. on Keypad)	keypadDecimal
Delete forward	deleteKey
Divide (/ on Keypad)	keypadDivide
End	endKey
Enter	enter
Escape	escape
Function keys	f1, f2, ...f35
Home	homeKey
Insert	insertKey
Minus (- on Keypad)	keypadMinus
Numlock	numLock
Option	OptionKey

To press this key...	Type this keyword (without quotation marks)
Page Down	pageDown
Page Up	pageUp
Pause/Break	pauseBreak
Plus (+ on Keypad)	keypadPlus
Printscreen	printScreen
Return	return
Scroll Lock	scrollLock
Shift	ShiftKey
Shift down	ShiftDown
Shift up	ShiftUp
Tab	tab
Times (* on Keypad)	keypadTimes
Windows	WindowsKey
Windows Alt	WindowsAltKey
Windows Alt down	WindowsAltDown
Windows Alt up	WindowsAltUp
All keys (keyUp command only)	AllKeys
All modifier keys (keyUp command only)	AllModifierKeys
All non-modifier keys (keyUp command only)	AllNonModifierKeys

## Appendix D: Global Properties

### EggPlant Global Properties

key	description
the ImageDoctor	Behavior of the Image Doctor during script execution; <i>Auto</i> , <i>Manual</i> , or <i>Off</i>
the InitialSuites	The first suite or suites that are searched for image or script resources during script execution.
the FinalSuites	The suite or suites that are searched for image or script resources after the script's own suite is searched.
the CommandLineOutput	Whether or not Log files are output when you run scripts from the command line
the CurrentTextPlatform	The current text platform
the DefaultTextStyle	The default text style of the current text platform
the DefaultUseMarkup	Whether or not supported text markups are used to format text images.
the RemoteClipboard	The contents of the SUT clipboard
the ScriptLogging	How much is written in the Log file
the ScriptAnimation	Whether script animation is turned on or off
the ScriptTracing	Whether script tracing is turned on or off
the SearchRectangle	The area that is searched in the Viewer window
the TextPlatforms	Property lists for all text platforms



## Run-Option Global Properties

key	description
the ForceScreenRefresh	Whether there is a screen refresh after every line
the ImageSearchCount	Number of times EggPlant searches for an image in the Viewer window
the ImageSearchDelay	Time between image searches
the ImageSearchTime	Time spent searching for an image
the KeyDownDelay	Time between key press and release
the MouseClickDelay	Time between mouse-button press and release
the MouseDoubleClickDelay	Time between clicks of a double-click
the MouseMoveDelay	Time between increments of mouse movement
the MouseMoveMode	Mouse path to the next given location
the MouseMoveSpeed	Space the mouse moves in each increment between locations
the NextKeyDelay	Time between keystrokes
the PreciseImageTolerance	Maximum color difference allowed by precise image searches
the ReadTextSettings	Default settings used by the ReadText and ReadTable functions
the RemoteWorkInterval	Minimum time between executed commands in the SUT
the RepositionPoint	The location where the mouse cursor is moved during searches
the SendShiftForCaps	Whether or not Shift is held down for capital letters on the SUT
the ShouldRepositionMouse	Whether the mouse is repositioned during searches
the StandardImageTolerance	Maximum color difference allowed by tolerant image searches

## Appendix E: Property Lists

### CaptureScreen Properties

key	description
Name	An image file name and optional path information, in quotation marks
Rectangle (or Rect)	A pair of diagonal locations indicating a rectangle to capture
Increment	Whether to append an automatically incremented number to the image; <i>true</i> or <i>false</i>
ImageInfo	An image property list

### Connection Properties

key	description
ServerID (required)	SUT host name, IP address, or display name
PortNum	Port number used by the SUT VNC server
Password	Password of the SUT VNC server
sshHost	Host name or IP address of an SSH host
sshUser	User account on the SSH host computer
sshPassword	Password of the user account on the SSH host computer
ColorDepth	The color depth of the SUT in the Viewer window: 8, 16, 32
Visible	Whether or not the Viewer window opens upon connection

## Image Properties

key	description
HotSpot	Coordinates of the Hot Spot within the image
ImageName	Name of the image file; (File extension is not used)
SearchType	<i>Precise, Tolerant, or Text</i>
Searchrectangle	Coordinates that define a rectangle in the Viewer window. EggPlant only looks for this image within the defined rectangle.
Pulsing	Whether the search type allows for pulsing; <i>true</i> or <i>false</i>
Tolerance	The acceptable difference between color values in an image and a match in the Viewer window
Discrepancy	The percentage or absolute number of pixels that may differ between an image and a match in the Viewer window
Cliptrectangle	Coordinates that define a rectangle within the image; any pixels outside the cliptrectangle are disregarded for image matching.
ImageDescription	The images description in the Suite Editor Images pane. (This value cannot be changed.)
ImageSize	Width and height in pixels. (This value cannot be changed.)
ImagePath	File path of the image file. (This value cannot be changed.)
CaptureLocation	Screen coordinates of the image when it was captured. (This value cannot be changed.)

## Results Properties

key	description
LogFile	The name and absolute path of the Log file
Errors	A count of errors logged for that run
Warnings	A count of the warnings logged for that run
Exceptions	A count of the EggPlant caught and uncaught exceptions raised
Duration	The length of time the script ran (or has been running) given in seconds
RunDate	The date and time the run was started
Status	The status of the run – <i>Success, Failure, or Running</i>
ReturnValue	The returned value of any return statements

## SendMail Properties

key	description
smtp_host (required)	The host name or IP Address of the mail server
smtp_type	The authentication scheme used by the mail server. On Mac OS X: <i>None</i> , <i>Plain</i> , <i>Login</i> , or <i>CRAM-MD5</i> . On Linux and Windows: <i>None</i> or <i>Plain</i> .
smtp_user	The user account on the mail server; used if an smtp_type is specified
smtp_password	The password for login to the mail server; used if an smtp_type is specified
To (required)	One or more addresses, separated by commas
From (required by some mail servers))	The user account sending the message
Subject	The subject line for the message
Body (or Message)	The text of the message
Attachment	A filename or list of filenames to attach to the e-mail
ReplyTo	The default address to which a reply is sent
CC	One or more addresses to which a copy of the message is sent, separated by commas
"Content-Type"	A mime type. For html e-mail, use the value "text/html". (The default is "text/plain".)

## TextPlatform Properties

key	description
Name	Name of the text platform
Engine	The text engine used by the platform
Generator	The text-image generator (TIG) used by the platform. (The Generator property is deprecated. Instead, use <i>Engine</i> .)
Styles	A property list for each of the platform's text styles

## Text Properties

key	description
<b>Common text properties (used in all text property lists)</b>	
Text	The text string that you want to find on the SUT. (Required.)
TextPlatform	The name of the platform on which your text will be found.
TextStyle	Either the name of a text style, or a list of text properties defined in a script
<b>Generated (TIG) text properties</b>	
TextFont	The name of the font used
TextSize	The size of the text in points
TextColor	The color of the text
TextBackgroundColor	The background color of the Text Image
Bold	Whether the font is displayed as bold; <i>true</i> or <i>false</i>
Italic	Whether the font is displayed in italics; <i>true</i> or <i>false</i>
Underline	Whether the text is underlined; <i>true</i> or <i>false</i>
HotSpot	Coordinates of the Hot Spot within the image
ImageName	Name of the Text Image file; (File extension is not used)
ImagePath	File path of the Text Image file
SearchType	<i>Precise</i> , <i>Tolerant</i> , or <i>Text</i>
Pulsing	Whether the search type allows for pulsing; <i>true</i> or <i>false</i>
CaptureLocation	Coordinates of the Text Image the last time it was found
ImageSize	Width and height in pixels
Trim	Whether or not extra space is trimmed away from around the text
Cache	Whether a cached Text Image is used, if available
Anti-aliasing	(Pango TIG only.) Whether text anti-aliasing is <i>on</i> or <i>off</i> .
UseMarkup	Whether supported markup tags are recognized as text attributes ( <i>on</i> ) or treated as string literals ( <i>off</i> ).
<b>Generic (OCR) text properties</b>	
CaseSensitive	Whether text searches consider case.
Contrast	Whether the SUT display is internally seen as a two-color image.
ContrastColor	The color that is considered the primary color when contrast is on.
ContrastTolerance	The maximum per-channel difference allowed or a pixel to be seen as the contrast color

## TextStyle Properties

key	description
<b>Generated (TIG) textStyle properties</b>	
TextFont	The name of the font used, in quotation marks
TextSize	The size of the text in points
TextColor	The color of the text
TextBackgroundColor	The background color of the Text Image
Bold	Whether the font is displayed as bold; <i>true</i> or <i>false</i>
Italic	Whether the font is displayed in italics; <i>true</i> or <i>false</i>
Underline	Whether the text is underlined; <i>true</i> or <i>false</i>
<b>Generic (OCR) textStyle properties</b>	
CaseSensitive	Whether text searches consider case.
Contrast	Whether the SUT display is internally seen as a two-color image
ContrastColor	The color that is considered the primary color when contrast is on
ContrastTolerance	The maximum per-channel difference allowed or a pixel to be seen as the contrast color

## Appendix F: Pango Markup Tags

### Pango Markup Tags (Currently Mac OS X only)

tag	attribute
b	Bold
big	Bigger
i	Italics
s	Strikethrough
sub	Subscript
sup	Superscript
small	Smaller
tt	Monospace font
u	Underline